



ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



Mô hình sinh dữ liệu

Hà Nội, 8/2022

Nội dung

1. Giới thiệu về mô hình sinh
2. Mô hình tự mã hóa Autoencoder
3. GANs

Giới thiệu về mạng sinh dữ liệu

Đâu là mặt thật, đâu là mặt giả?

- <http://www.whichfaceisreal.com/>



Học giám sát và không giám sát

Học giám sát

- Dữ liệu: (x, y)
 x là dữ liệu, y là nhãn
- Mục đích: Học hàm số để ánh xạ $x \rightarrow y$
- Ví dụ: Phân loại, hồi quy, phát hiện đối tượng, phân đoạn ngữ nghĩa, dịch máy...

Học không giám sát

- Dữ liệu: x
 x là dữ liệu, không nhãn!
- Mục đích: Học một cấu trúc ẩn hay một cấu trúc nền tảng nào đó của dữ liệu
- Ví dụ: phân cụm, giảm chiều...

Mô hình sinh

- Mục đích: Nhận đầu vào một tập mẫu huấn luyện sinh ra từ một phân bố nào đó và học một mô hình để có thể biểu diễn lại phân bố đó



- Làm sao để học $p_{model}(x)$ tương tự với $p_{data}(x)$?

Tại sao cần mô hình sinh?



Homogeneous skin color, pose

VS

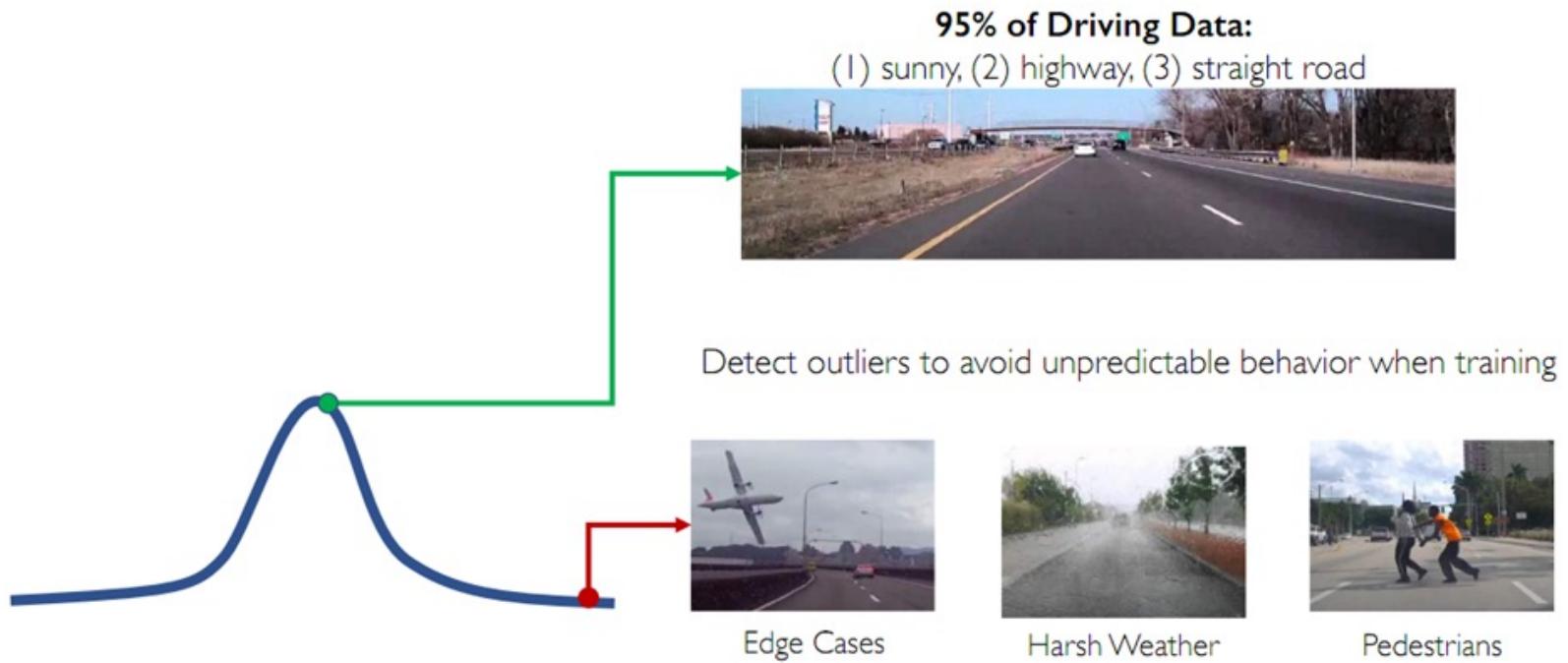


Diverse skin color, pose, illumination

- Sử dụng phân bố ẩn học được để sinh ra dữ liệu đa dạng và cân bằng hơn (debias)

Tại sao cần mô hình sinh?

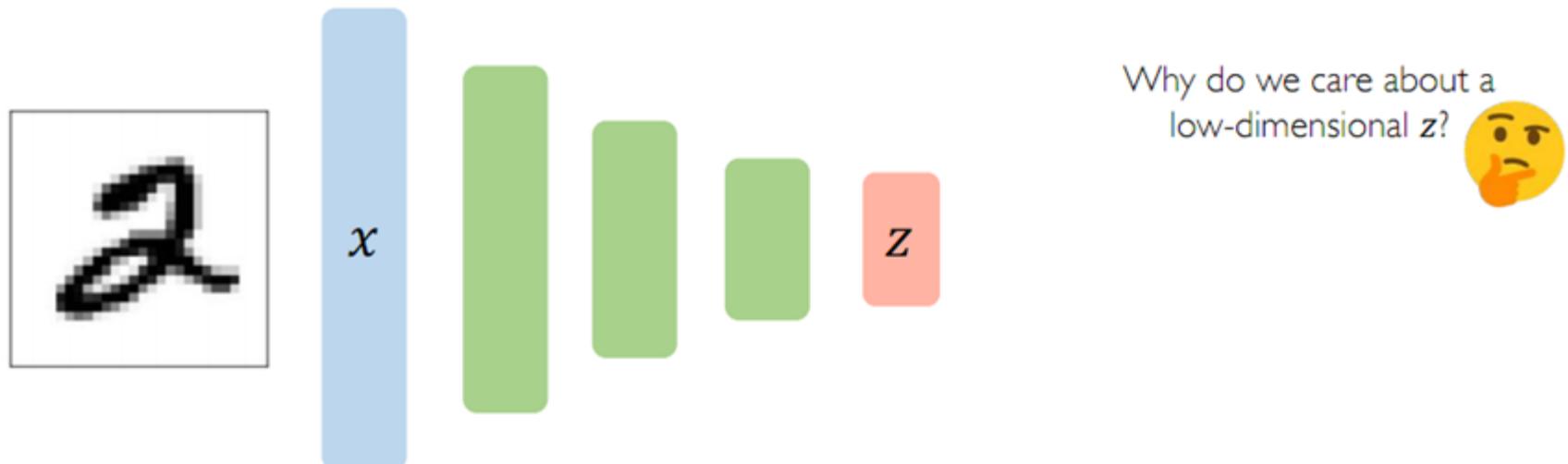
- Phát hiện ngoại lệ (outlier): Làm sao để phát hiện một sự kiện mới học hiếm xảy ra?
- Sử dụng mô hình sinh để học phân bố dữ liệu, từ đó xác định ngoại lệ dựa trên phân bố học được.



Mô hình tự mã hóa Autoencoder

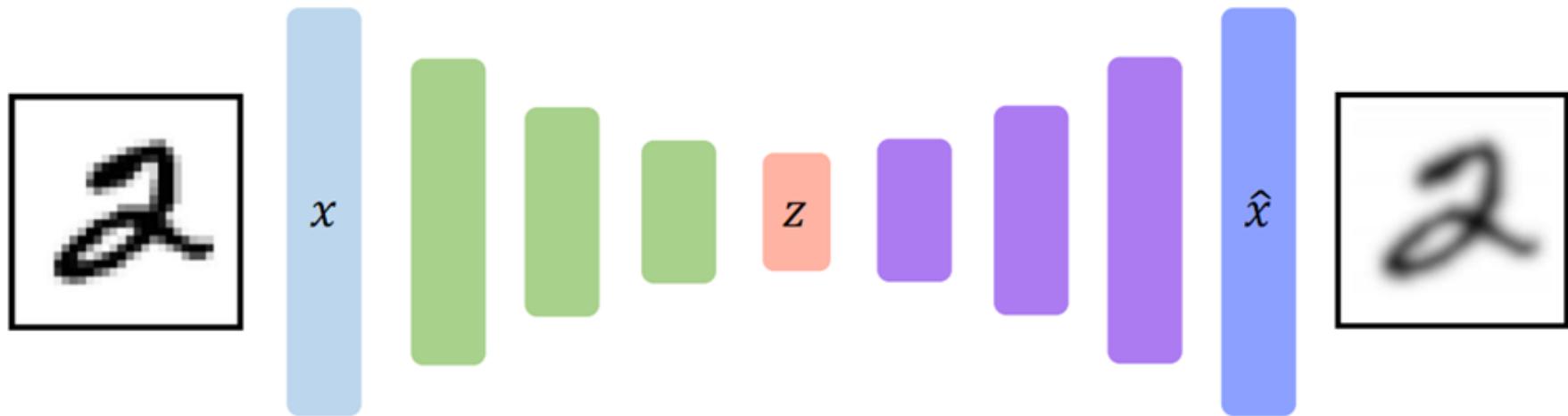
Autoencoder

- Là mô hình không giám sát cho phép học biểu diễn đặc trưng với số chiều nhỏ hơn từ tập huấn luyện không có nhãn
- “Encoder” học ánh xạ từ dữ liệu x vào không gian ẩn z có số chiều thấp hơn



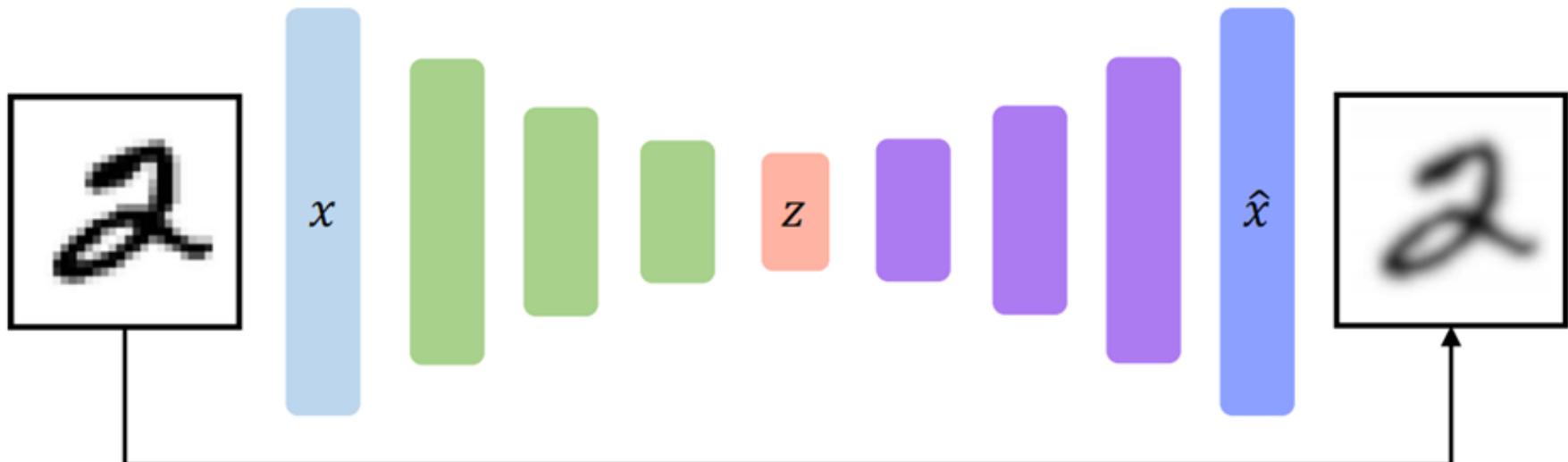
Autoencoder

- Làm sao để học không gian ẩn?
- Huấn luyện mô hình sử dụng đặc trưng ẩn z để khôi phục lại dữ liệu gốc ban đầu
- “Decoder” ánh xạ đặc trưng ẩn z ngược trở lại để khôi phục thông tin dữ liệu đầu vào \hat{x}



Autoencoder

- Làm sao để học không gian ẩn?
- Huấn luyện mô hình sử dụng đặc trưng ẩn z để khôi phục lại dữ liệu gốc ban đầu
- Hàm mục tiêu không cần nhãn!



$$\mathcal{L}(x, \hat{x}) = \|x - \hat{x}\|^2$$

Số chiều không gian ẩn ảnh hưởng chất lượng khôi phục dữ liệu

- Autoencoder là một kiểu nén dữ liệu.
- Số chiều không gian ẩn càng nhỏ càng tạo ra nút thắt cổ chai (bottleneck) lớn khi huấn luyện

2D latent space



5D latent space



Ground Truth

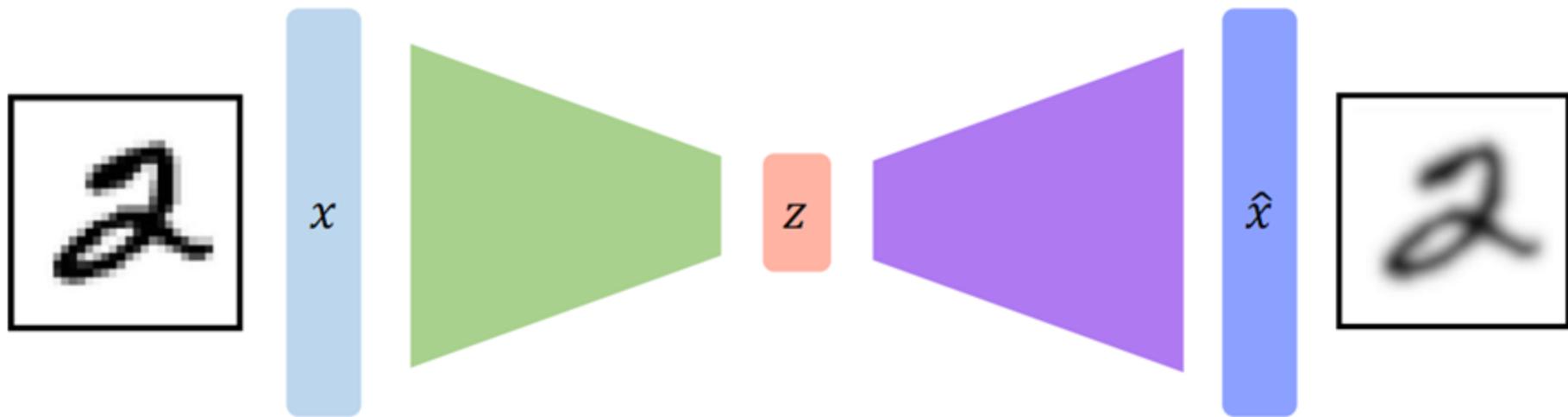


Autoencoders để học biểu diễn

- Các lớp ẩn thắt cổ chai ép mạng học biểu diễn ẩn nén thông tin dữ liệu vào
- Hàm mục tiêu tái tạo ép biểu diễn ẩn phải mã hóa được càng nhiều thông tin từ dữ liệu vào càng tốt
- **Autoencoding = Automatically encoding data**

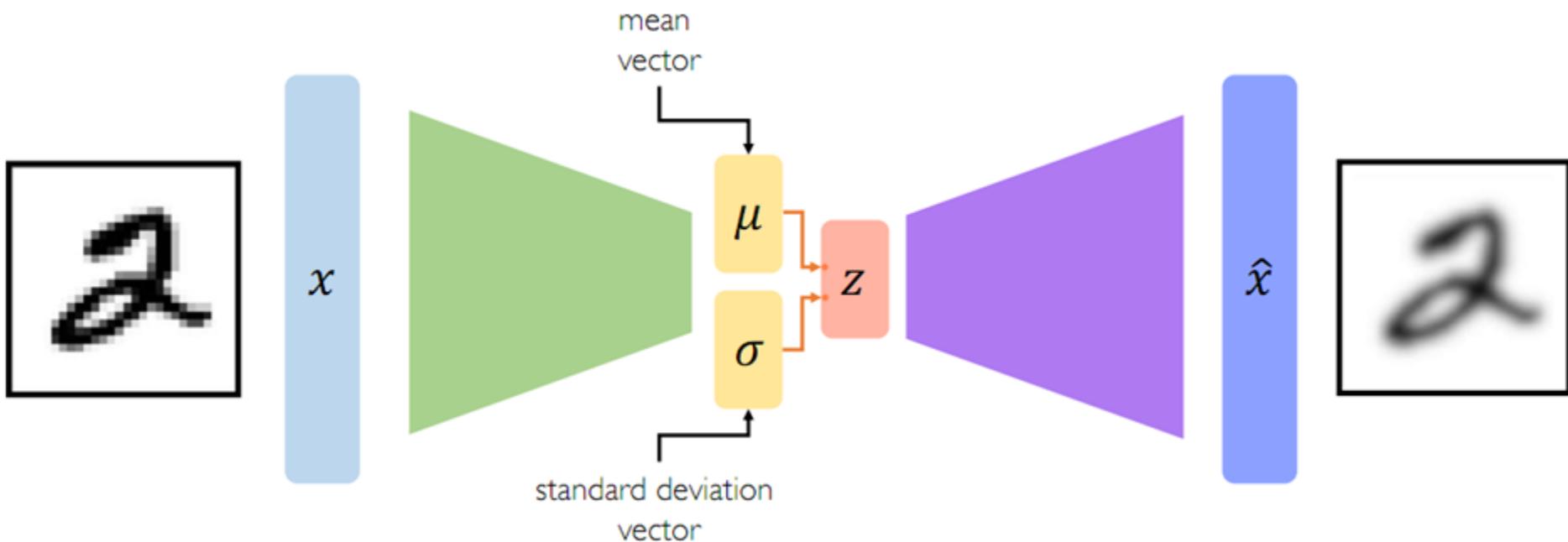
Variational Autoencoders

- VAEs: sự khác biệt chính so với autoencoder truyền thống

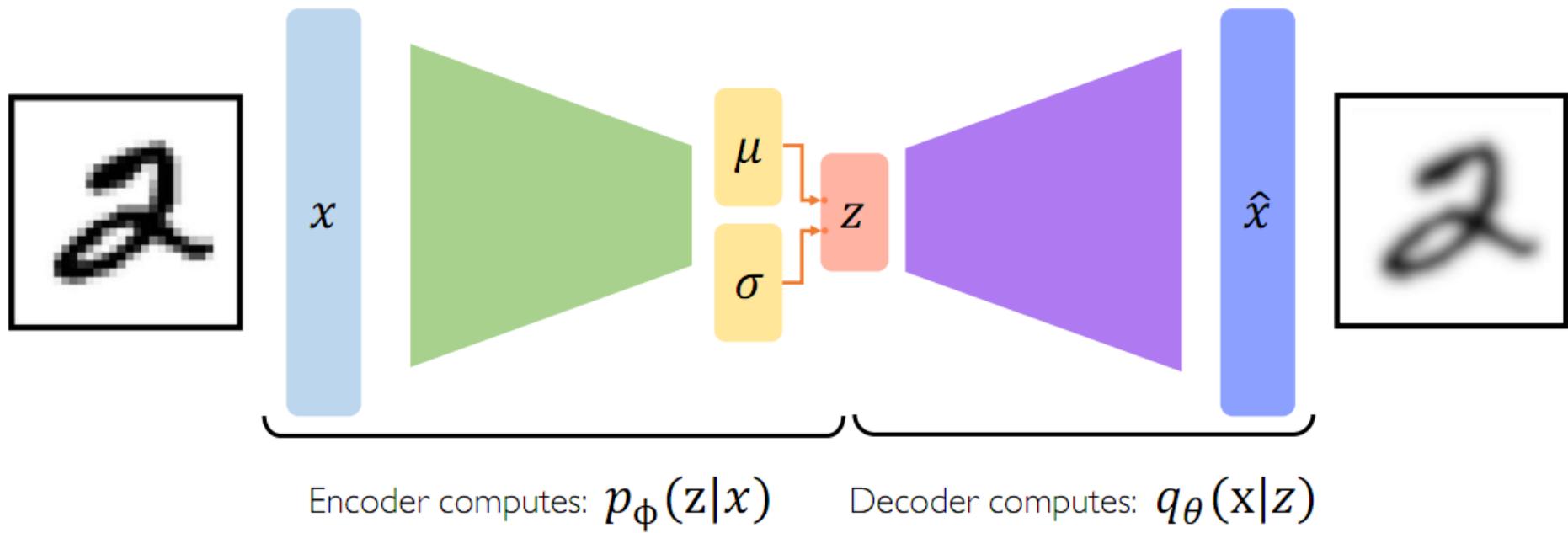


Variational Autoencoders

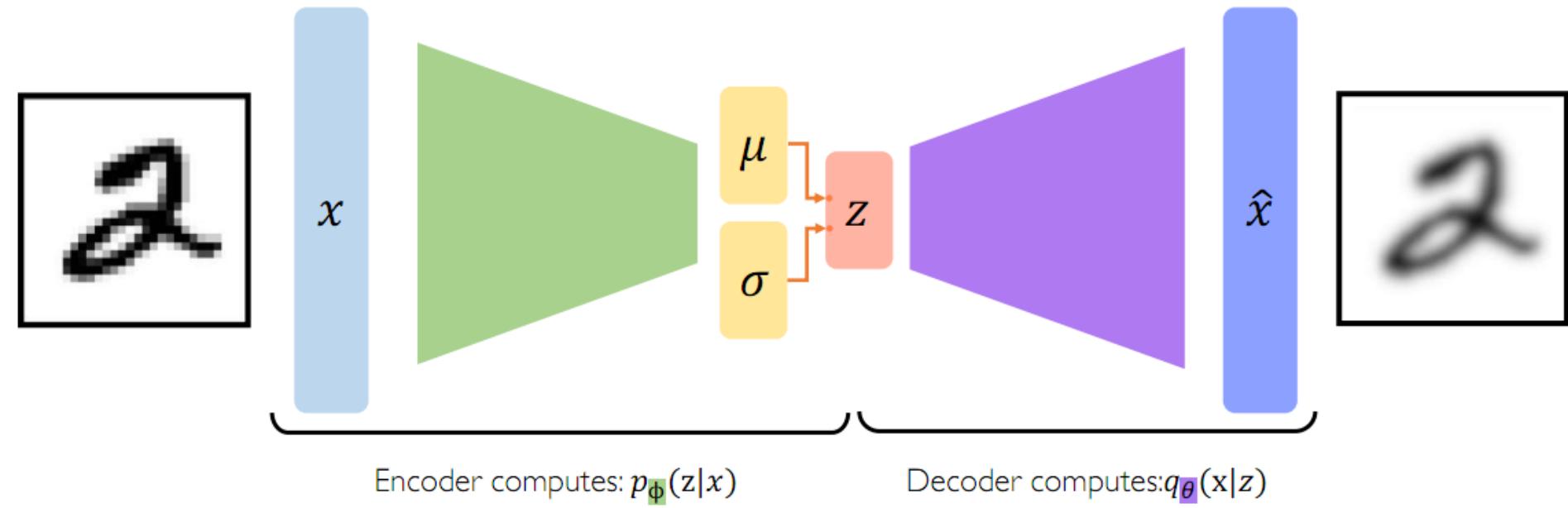
- VAEs: sự khác biệt chính so với autoencoder truyền thống



Tối ưu VAE

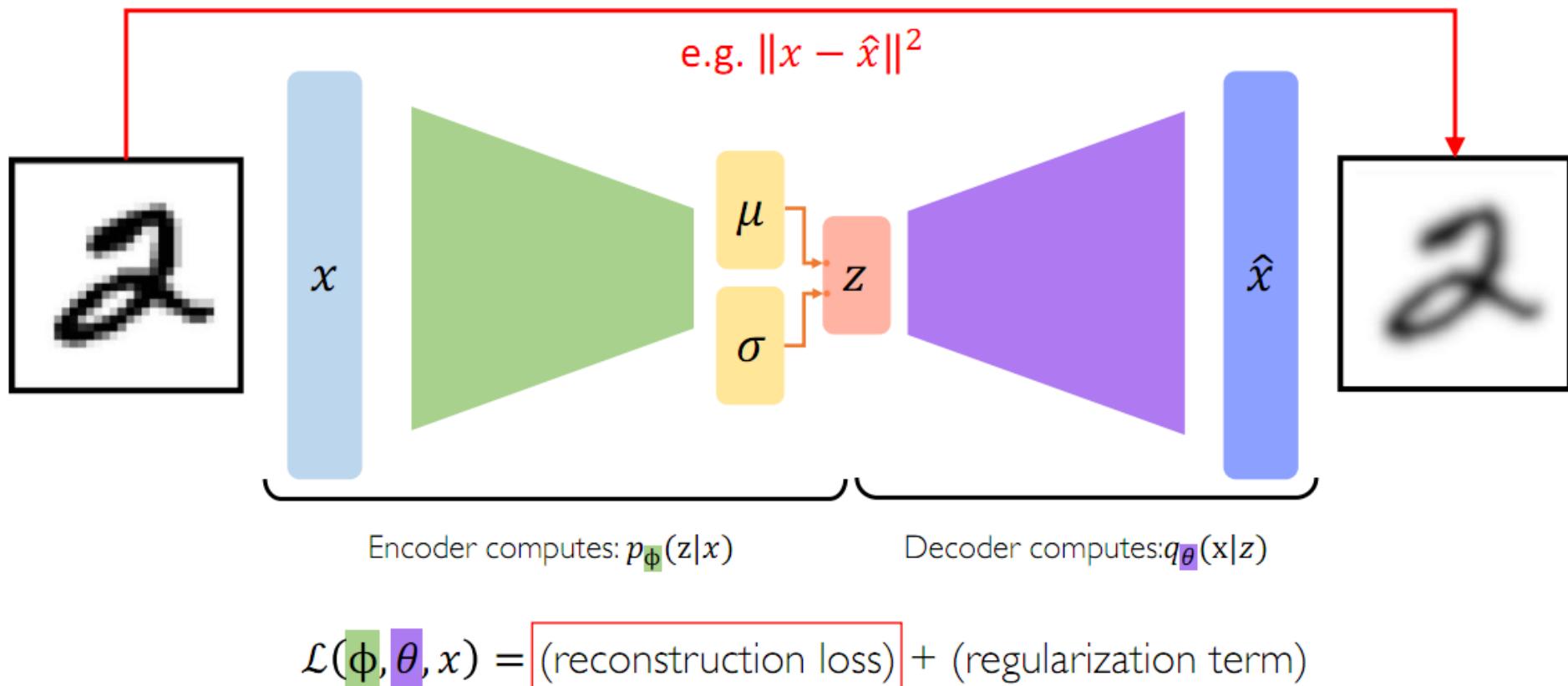


Tối ưu VAE

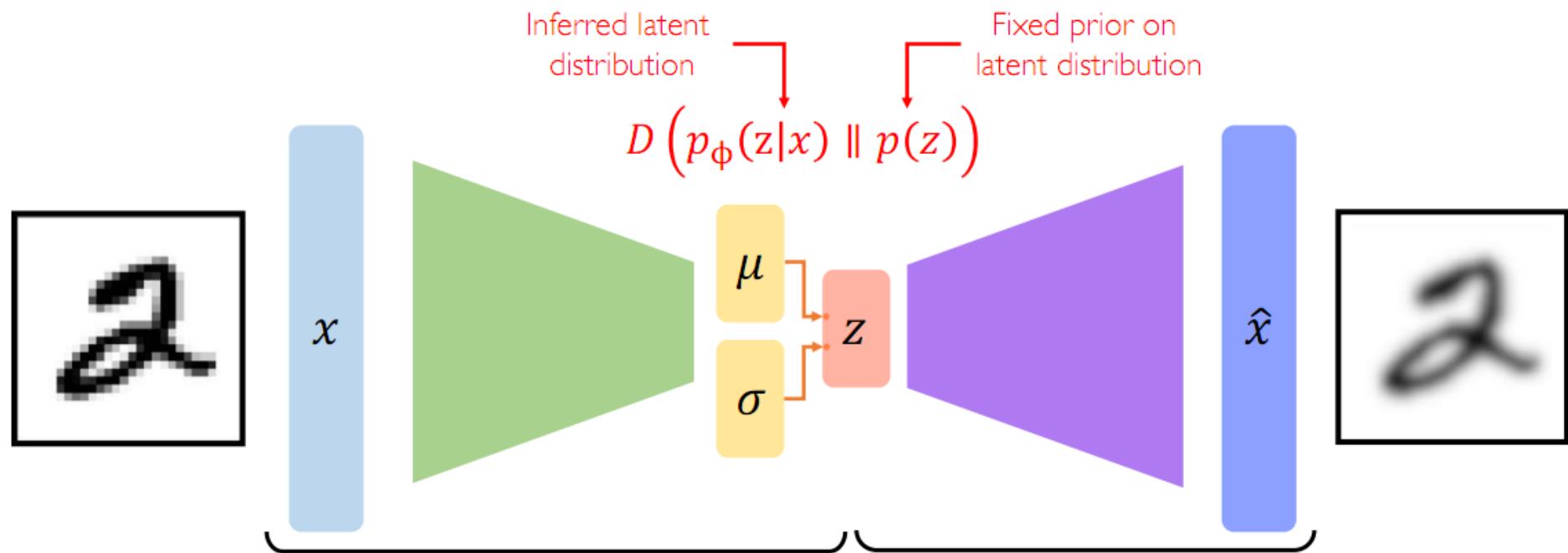


$$\mathcal{L}(\phi, \theta, x) = \text{(reconstruction loss)} + \text{(regularization term)}$$

Tối ưu VAE



Tối ưu VAE

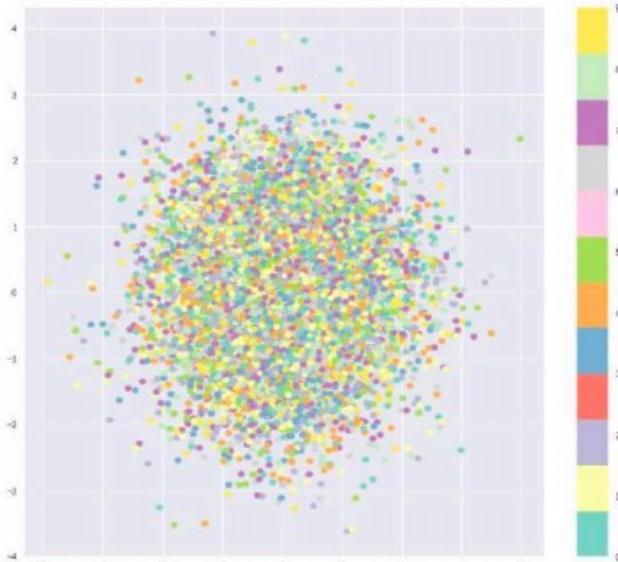


$$\mathcal{L}(\phi, \theta, x) = (\text{reconstruction loss}) + \boxed{(\text{regularization term})}$$

Phân bố tiên nghiệm của không gian ẩn

$$D(p_{\phi}(z|x) \parallel p(z))$$

↑ ↑
Inferred latent Fixed prior on
distribution latent distribution



Common choice of prior:

$$p(z) = \mathcal{N}(\mu = 0, \sigma^2 = 1)$$

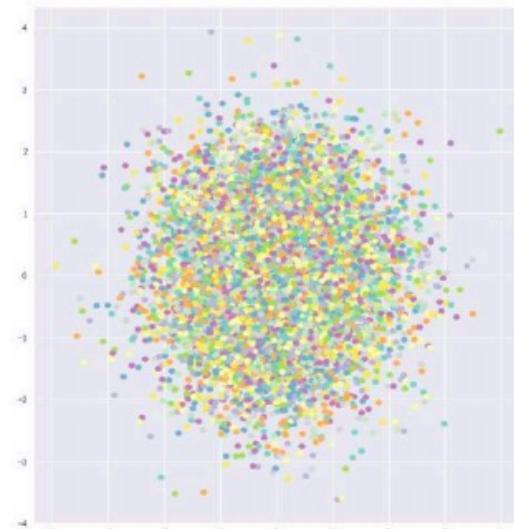
- Encourages encodings to distribute encodings evenly around the center of the latent space
 - Penalize the network when it tries to “cheat” by clustering points in specific regions (ie. memorizing the data)

Phân bố tiên nghiệm của không gian ẩn

$$D(p_{\phi}(z|x) \parallel p(z))$$

$$= -\frac{1}{2} \sum_{j=0}^{k-1} (\sigma_j + \mu_j^2 - 1 - \log \sigma_j)$$

KL-divergence between
the two distributions

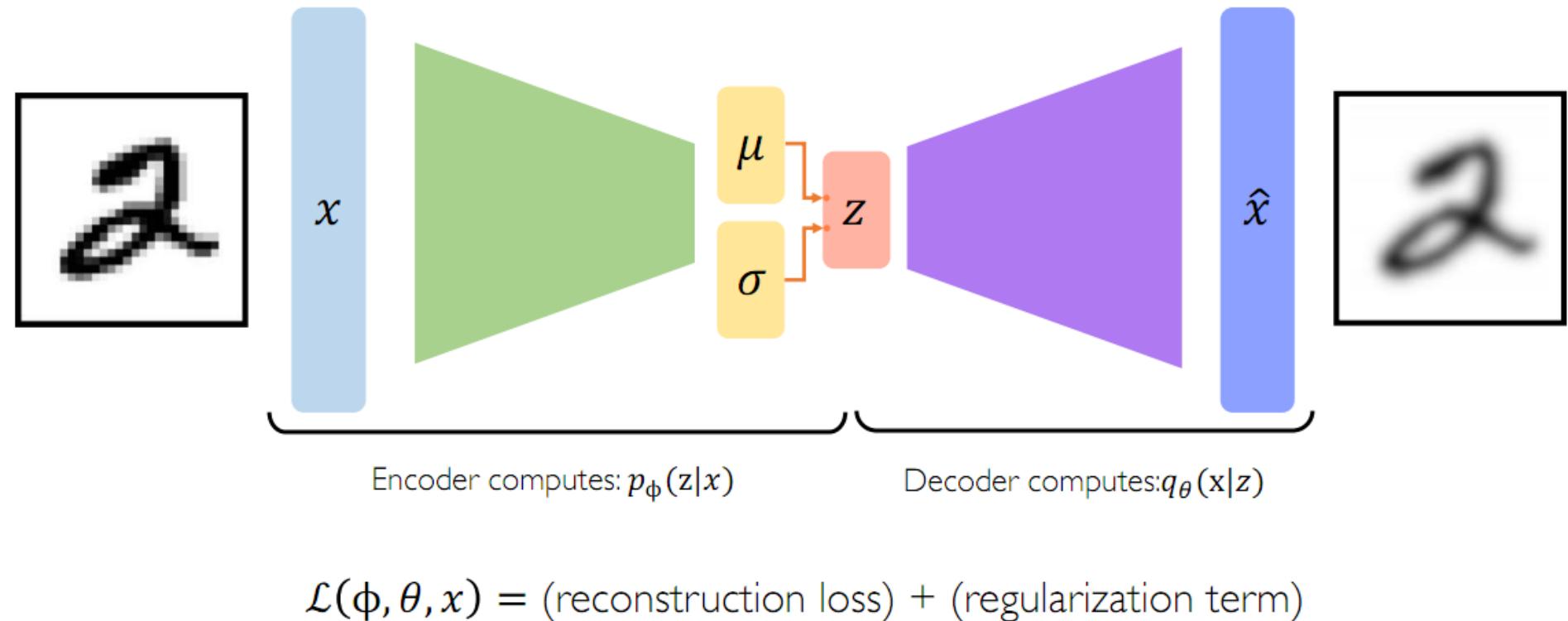


Common choice of prior:

$$p(z) = \mathcal{N}(\mu = 0, \sigma^2 = 1)$$

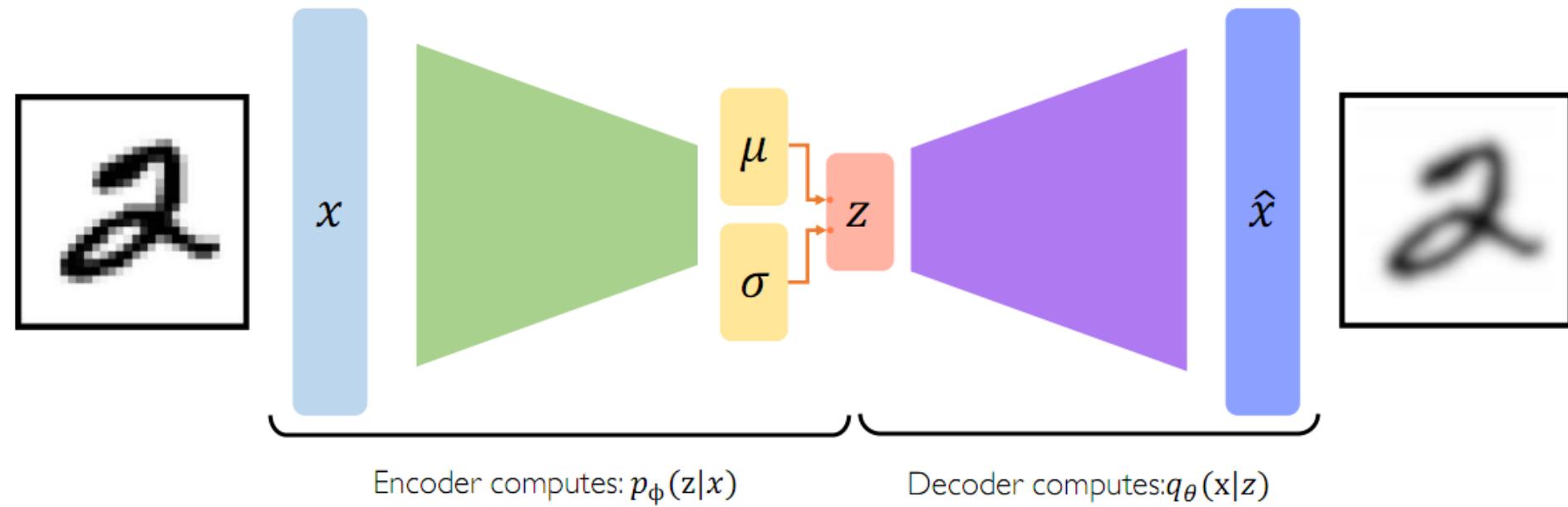
- Encourages encodings to distribute evenly around the center of the latent space
- Penalize the network when it tries to "cheat" by clustering points in specific regions (ie. memorizing the data)

Đồ thị tính toán của VAE



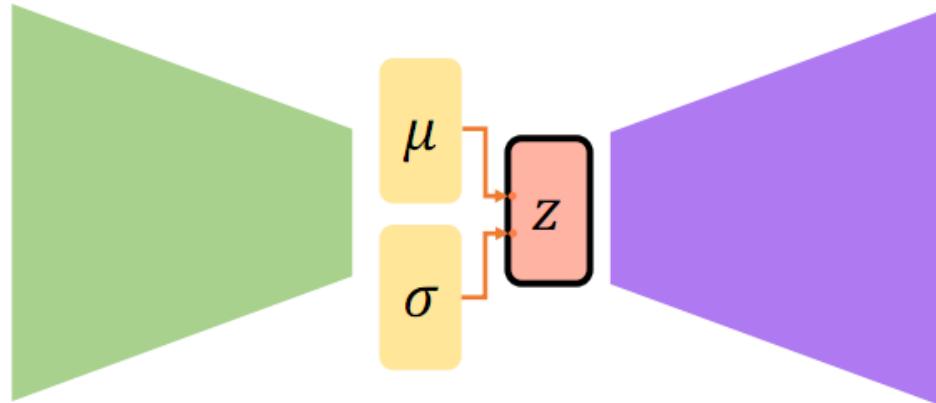
Đồ thị tính toán của VAE

- Vấn đề: Không thể lan truyền ngược qua lớp lấy mẫu!



$$\mathcal{L}(\phi, \theta, x) = (\text{reconstruction loss}) + (\text{regularization term})$$

Tái tham số hóa lớp lấy mẫu



Key Idea:

$$z \sim \mathcal{N}(\mu, \sigma^2)$$

Consider the sampled latent vector as a sum of

- a fixed μ vector,
- and fixed σ vector; scaled by random constants drawn from the prior distribution

$$\Rightarrow z = \mu + \sigma \odot \varepsilon$$

where $\varepsilon \sim \mathcal{N}(0,1)$

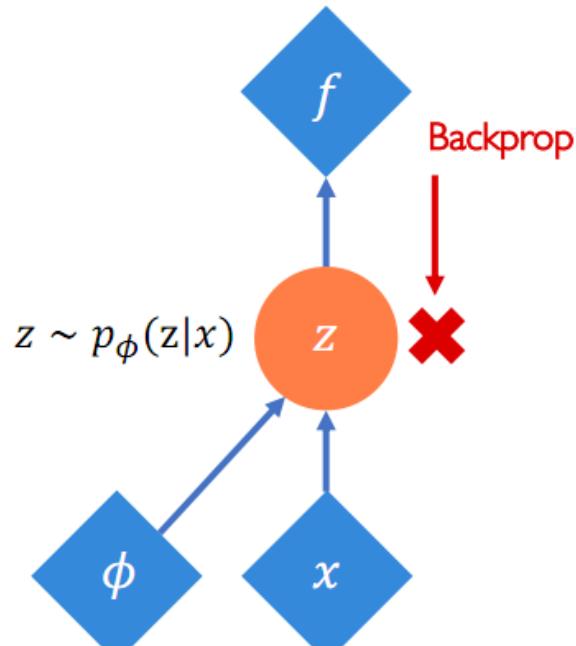
Tái tham số hóa lớp lấy mẫu



Deterministic node

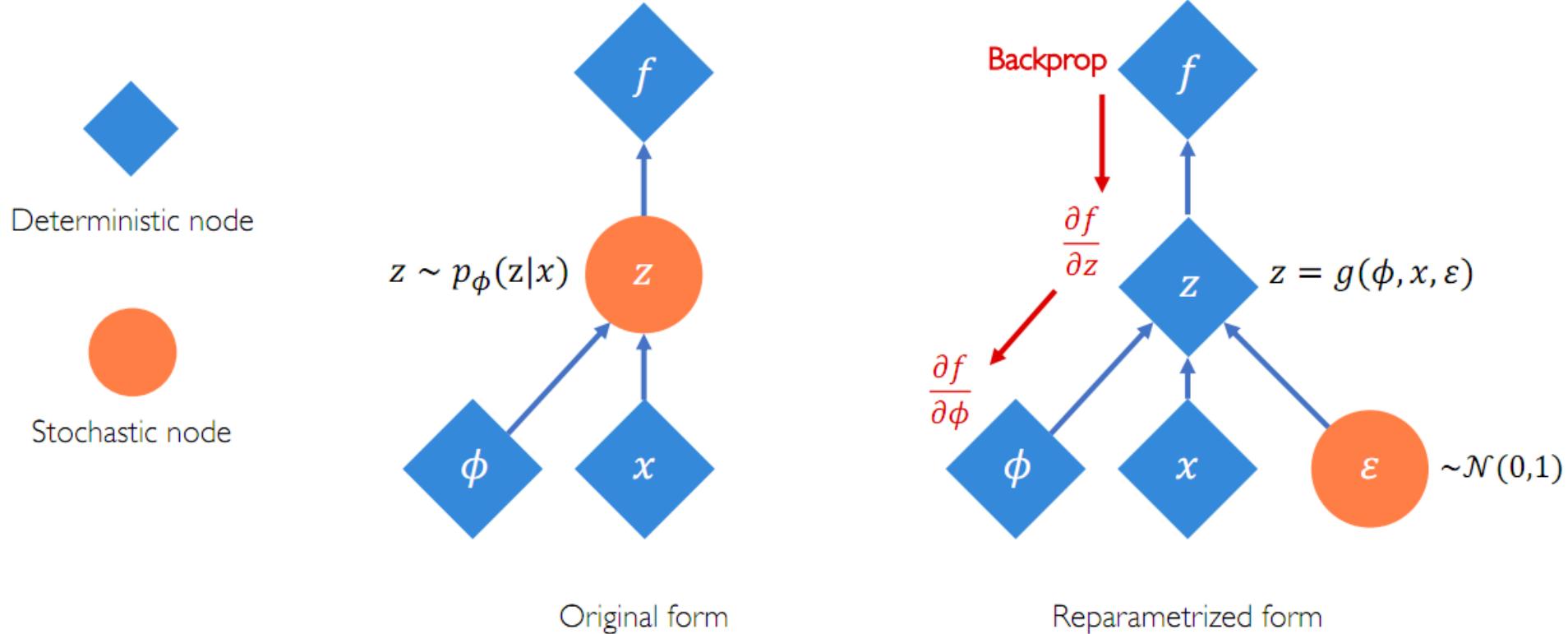


Stochastic node



Original form

Tái tham số hóa lớp lấy mẫu



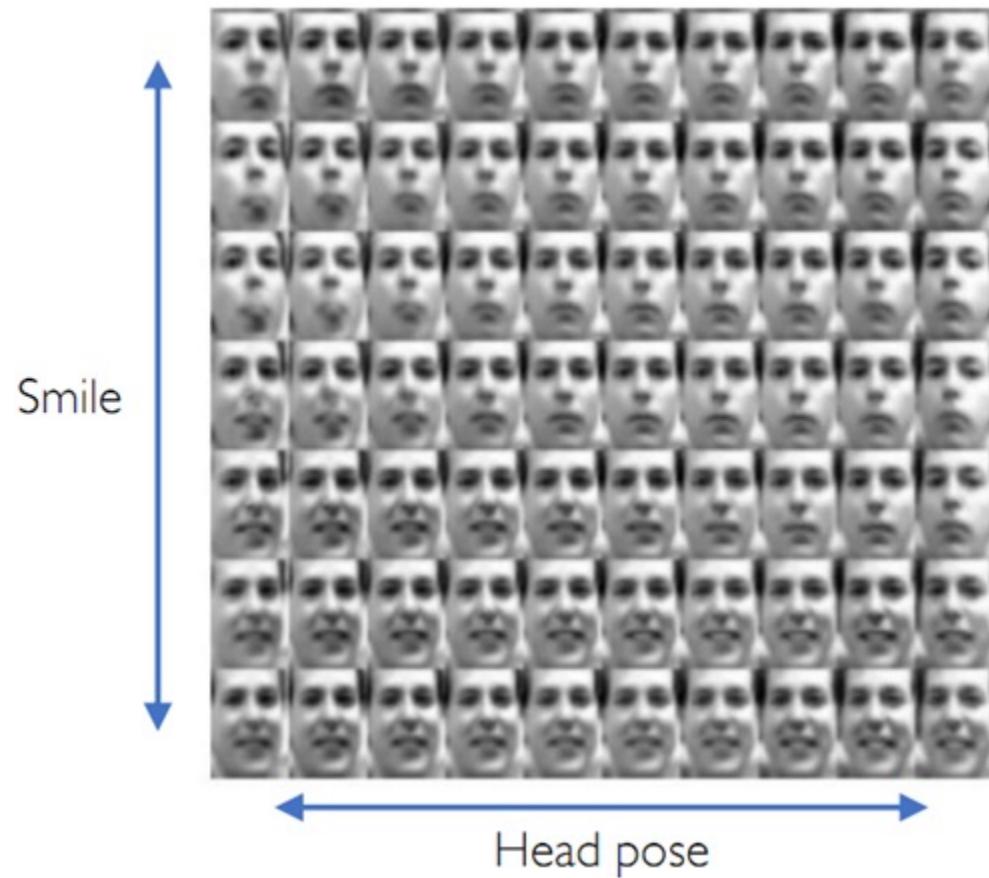
VAEs: Biến đổi không gian ẩn

- Tăng hoặc giảm từ từ một biến ẩn, giữ các biến khác cố định
- Mỗi chiều của z mã hóa các đặc trưng ẩn có ý nghĩa khác nhau



VAEs: Biến đổi không gian ẩn

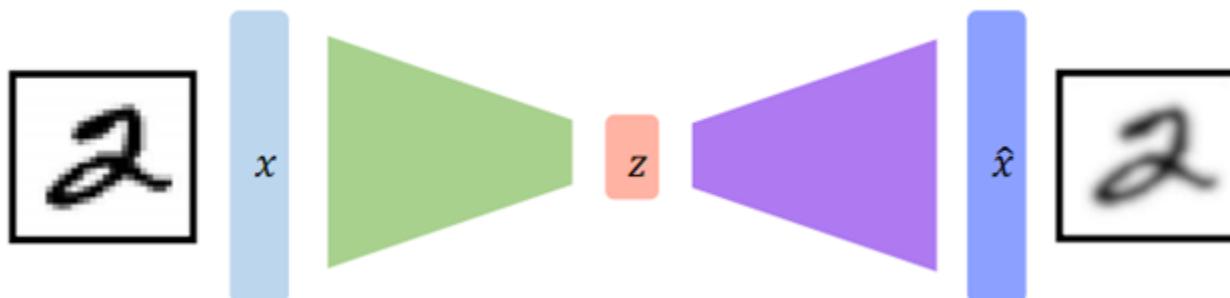
- Lý tưởng là các biến ẩn hoàn toàn độc lập với nhau (uncorrelated)
- Có thể thêm ràng buộc dạng ma trận chéo để ép các biến ẩn độc lập với nhau (hệ số tương quan giữa các biến ẩn khác nhau xấp xỉ 0)



Disentanglement

VAEs: Tổng kết

1. Hàm mục tiêu tái tạo cho phép huấn luyện không cần nhãn (không giám sát)
2. Sử dụng kỹ thuật tái tham số hóa để cho phép huấn luyện end-to-end
3. Diễn giải các biến ẩn bằng cách biến đổi giá trị của nó và quan sát
4. Sinh dữ liệu mới

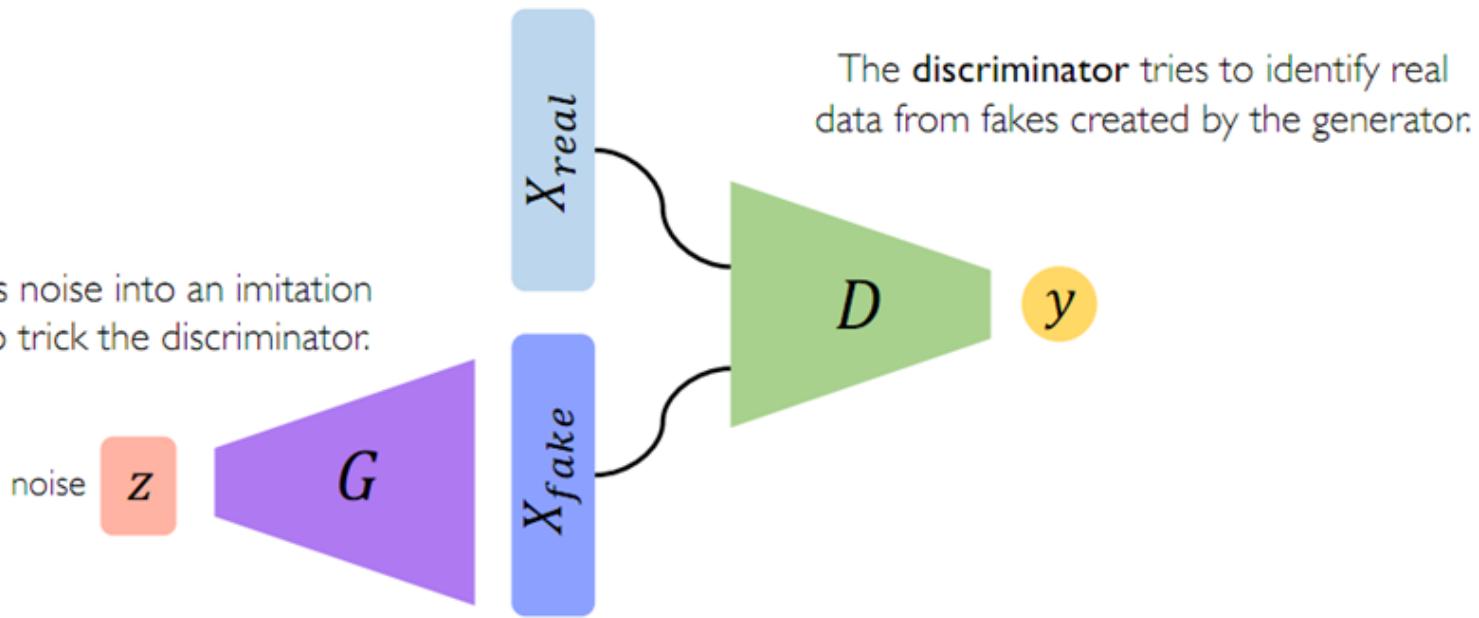


GANs

Mạng sinh dữ liệu

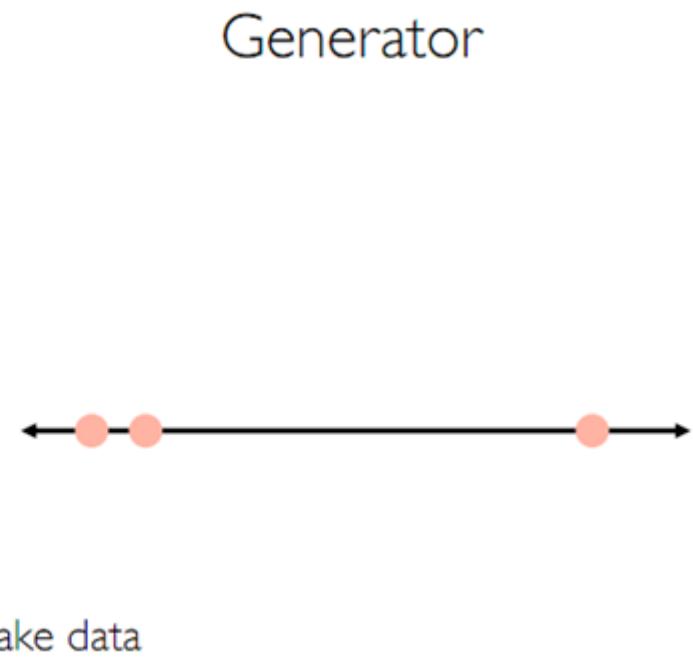
- GANs là một mô hình sinh chứa hai mạng nơ-ron đối chơi lẫn nhau
- Mạng sinh (generator) biến véctơ nhiễu thành một dữ liệu giả để đánh lừa mạng phân loại (discriminator)

The generator turns noise into an imitation of the data to try to trick the discriminator.



Trực giác ban đầu về GANs

- Generator tạo dữ liệu giả từ nhiễu



Trực giác ban đầu về GANs

- Discriminator nhìn vào dữ liệu thật và giả



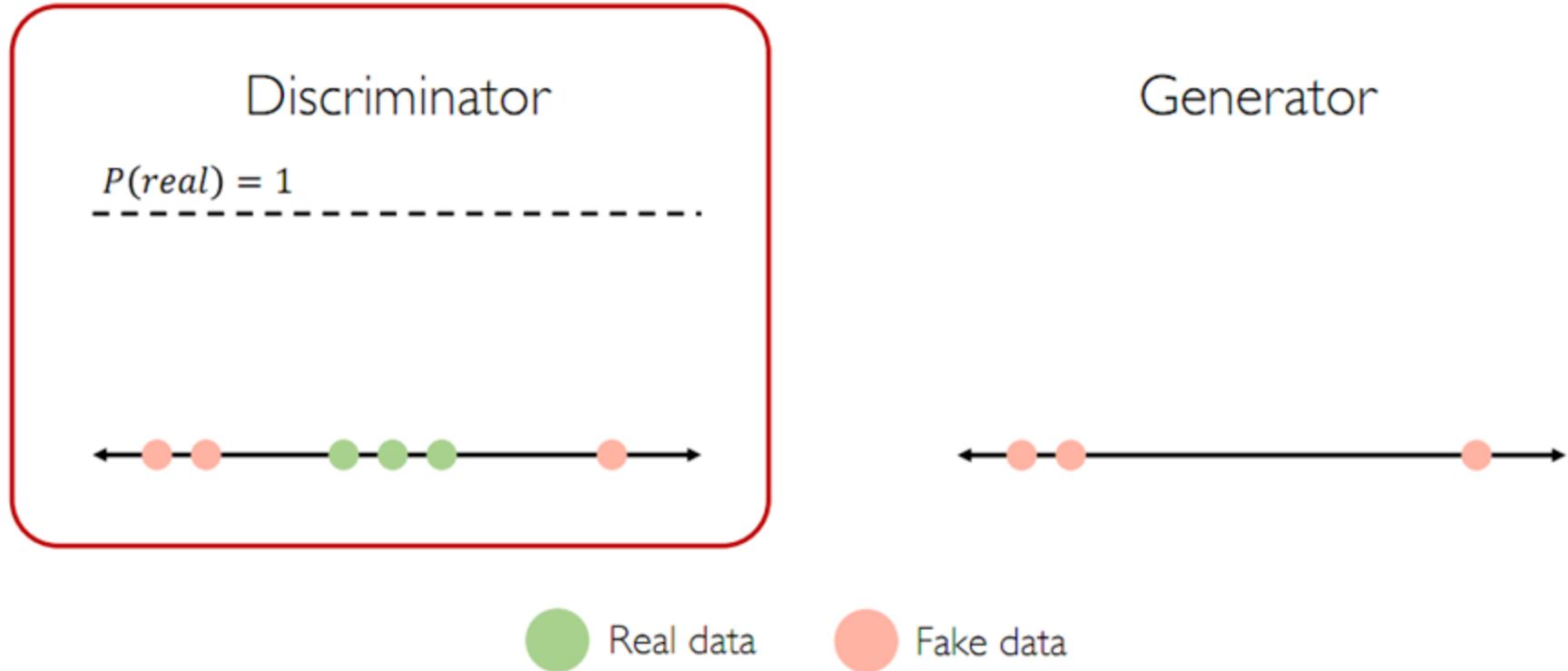
Trực giác ban đầu về GANs

- Discriminator nhìn vào dữ liệu thật và giả



Trực giác ban đầu về GANs

- Discriminator đoán đâu là thật, đâu là giả



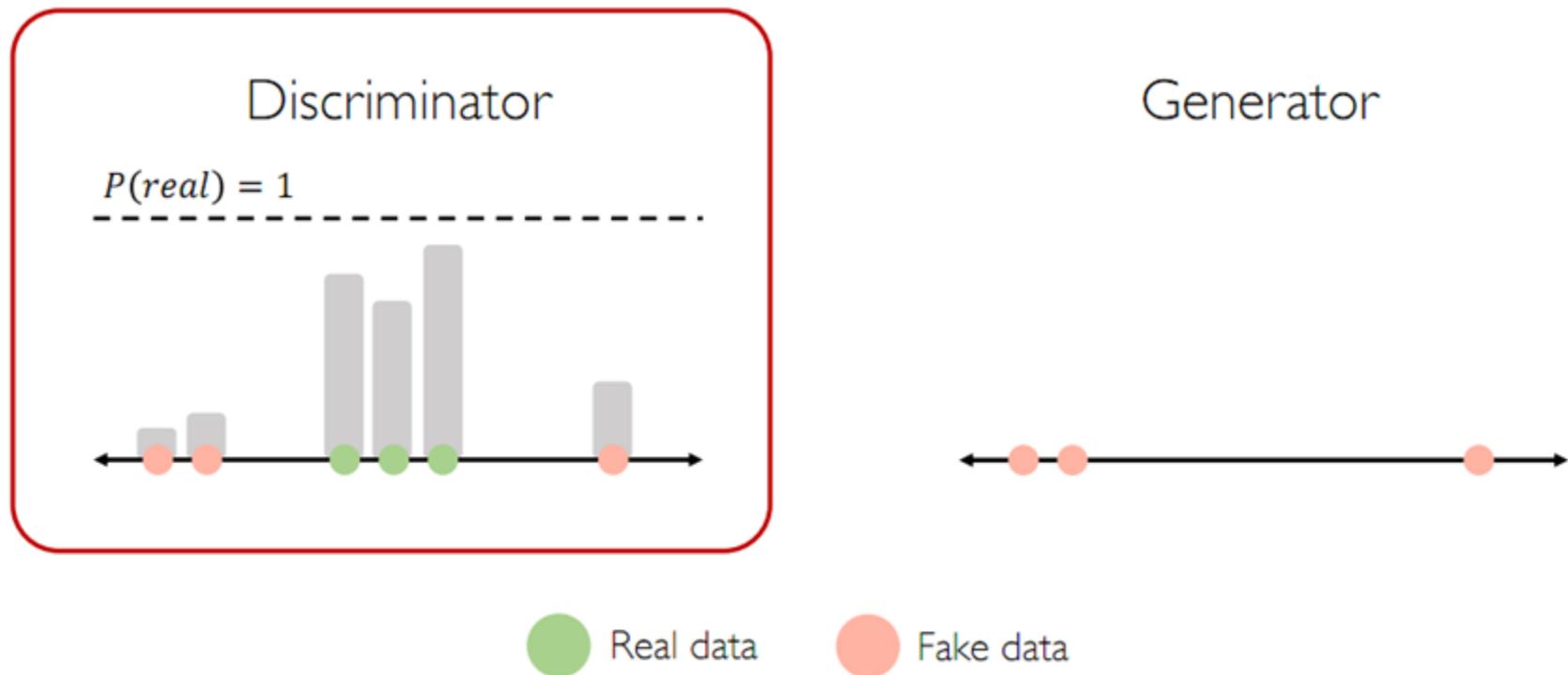
Trực giác ban đầu về GANs

- Discriminator đoán đâu là thật, đâu là giả



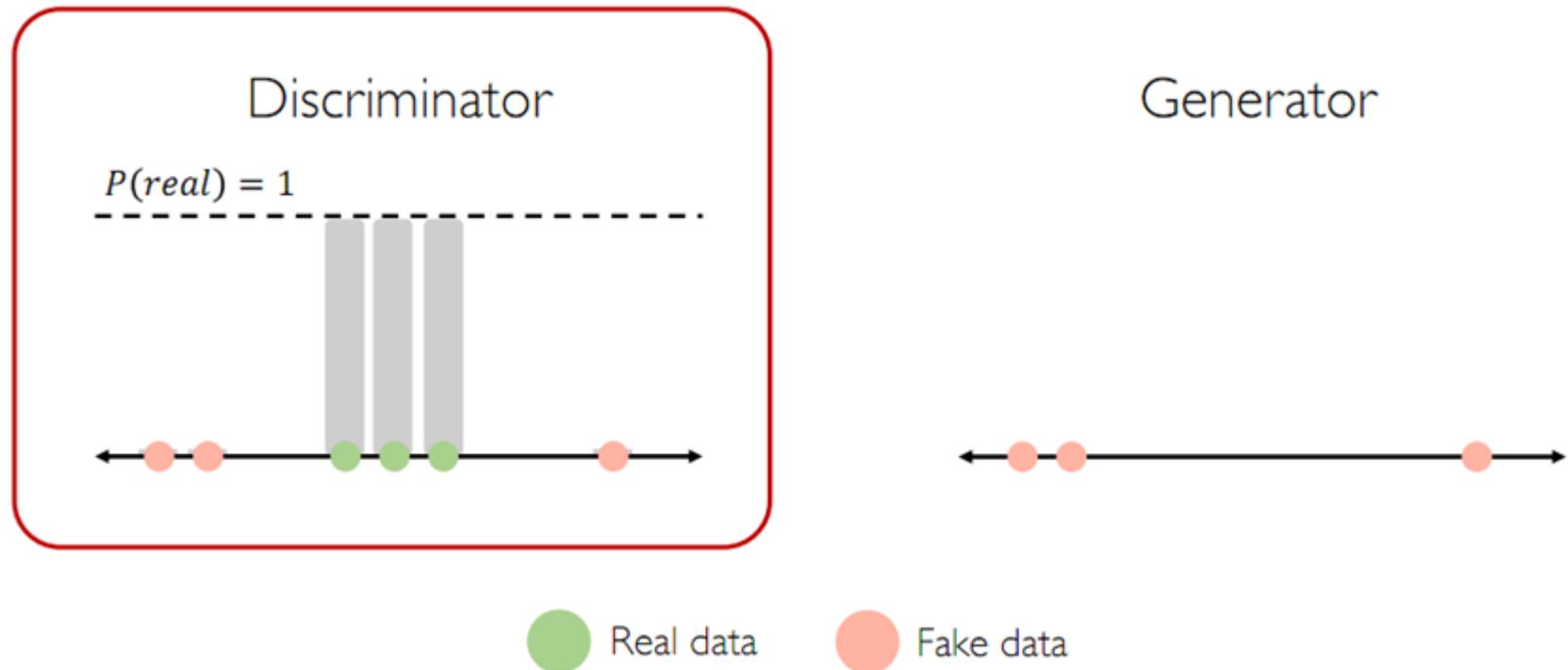
Trực giác ban đầu về GANs

- Discriminator đoán đâu là thật, đâu là giả



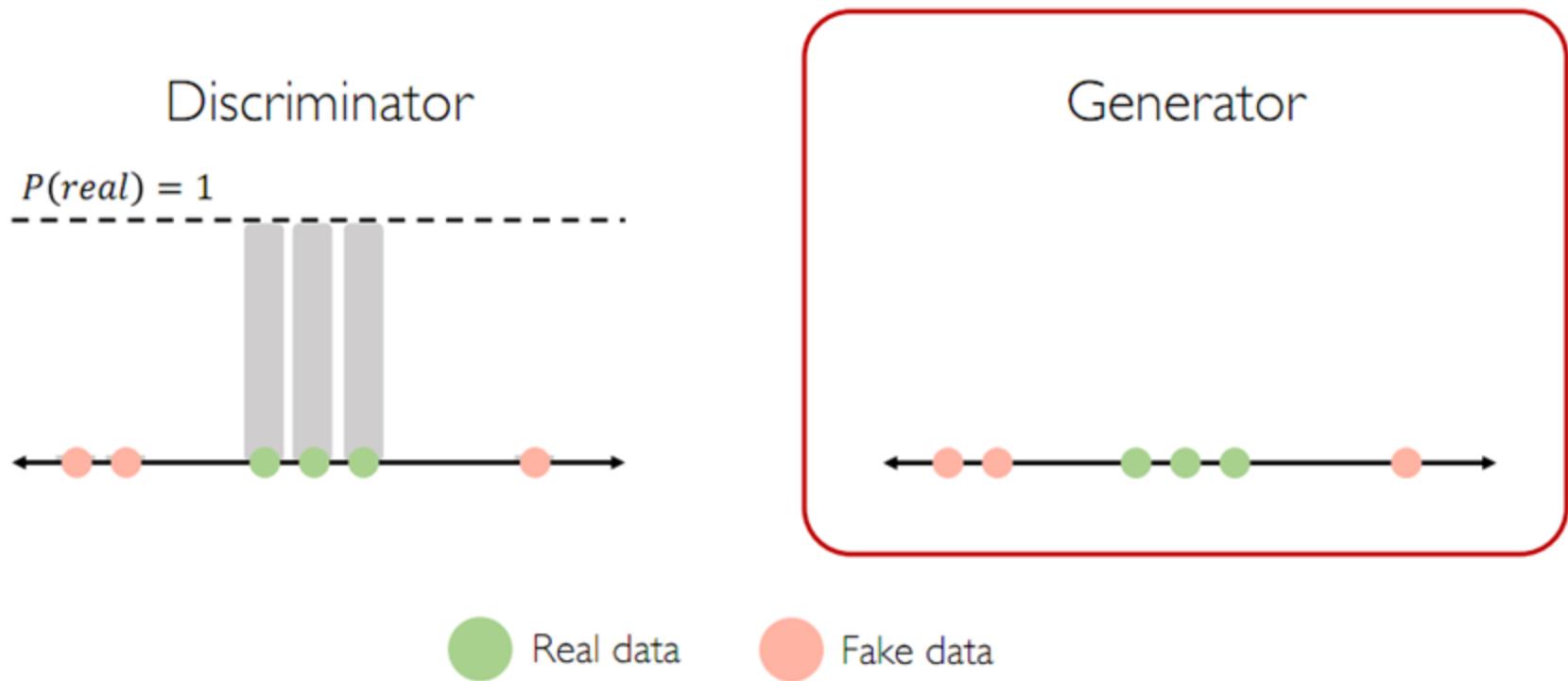
Trực giác ban đầu về GANs

- Discriminator đoán đâu là thật, đâu là giả



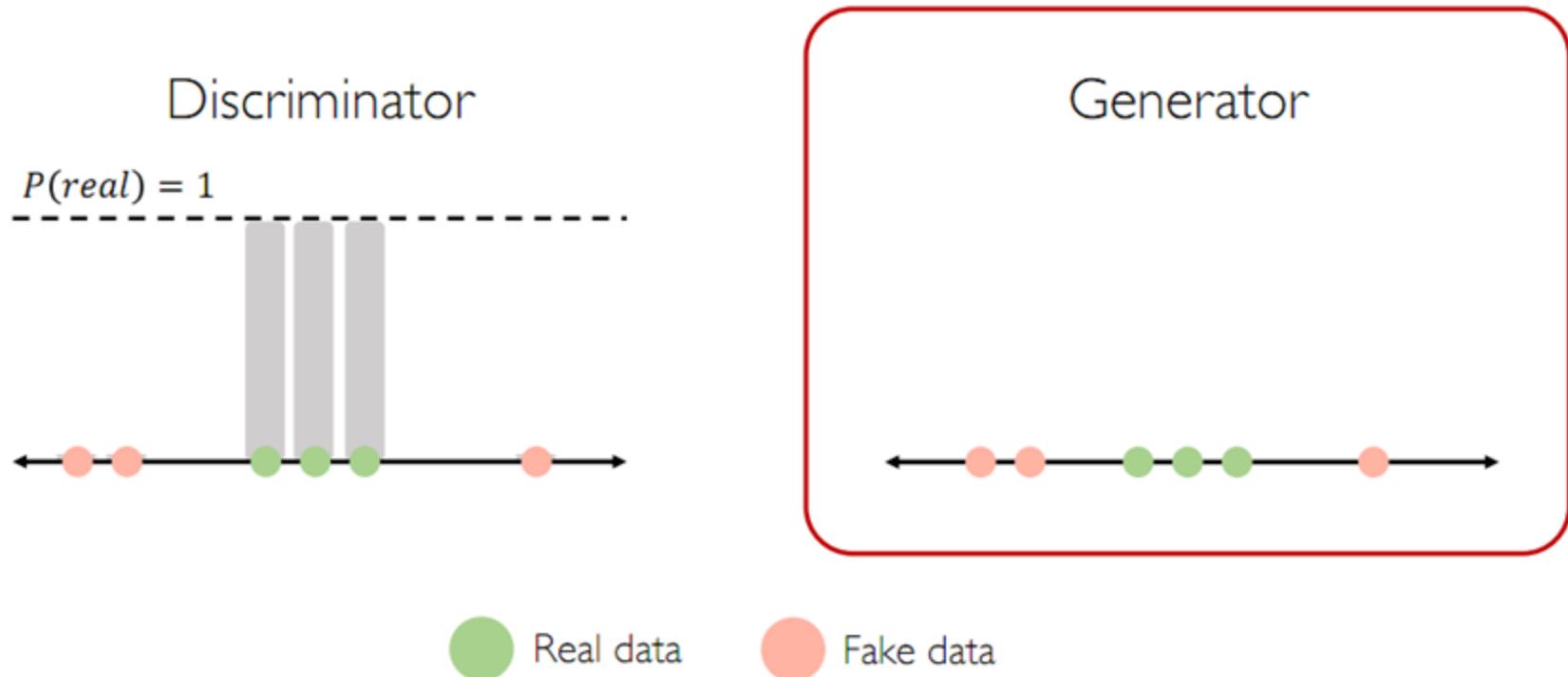
Trực giác ban đầu về GANs

- Generator cố gắng cải tiến chất lượng dữ liệu giả



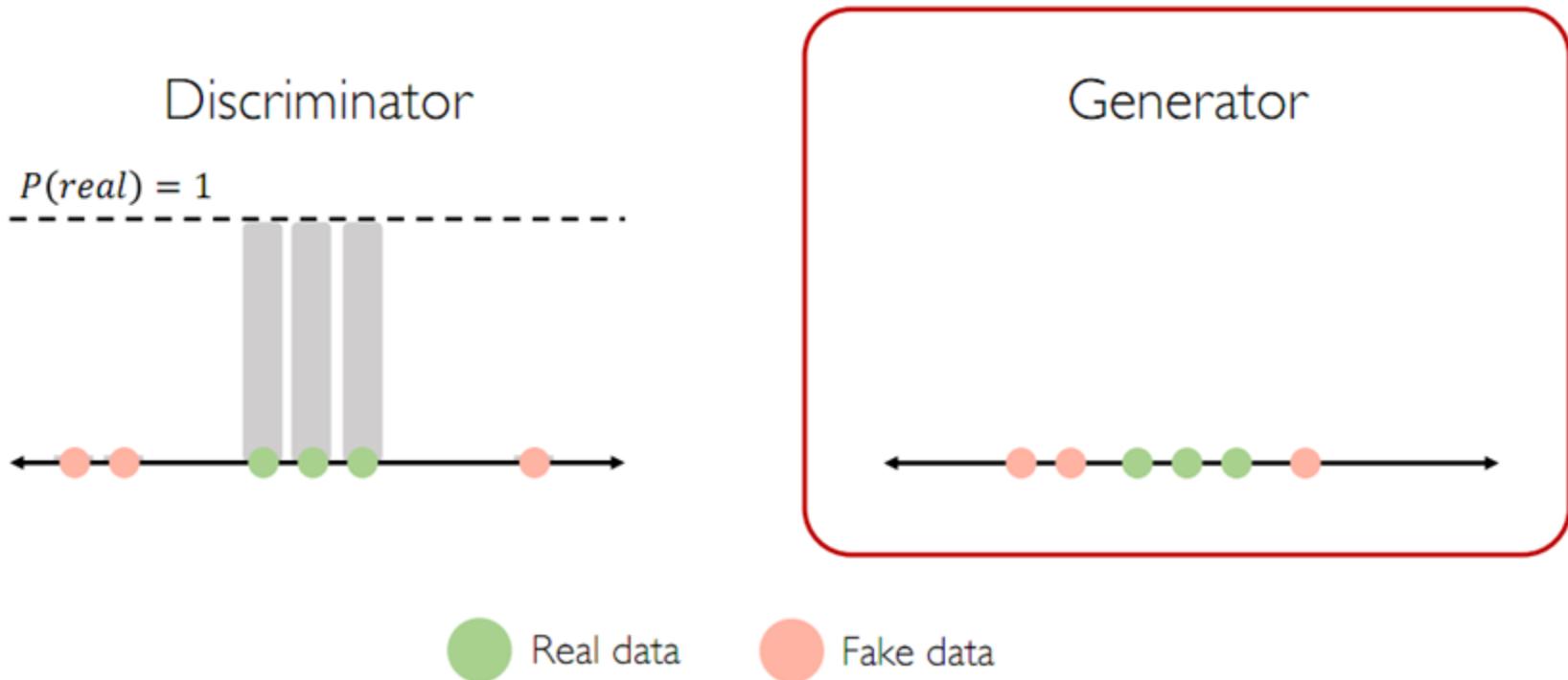
Trực giác ban đầu về GANs

- Generator cố gắng cải tiến chất lượng dữ liệu giả



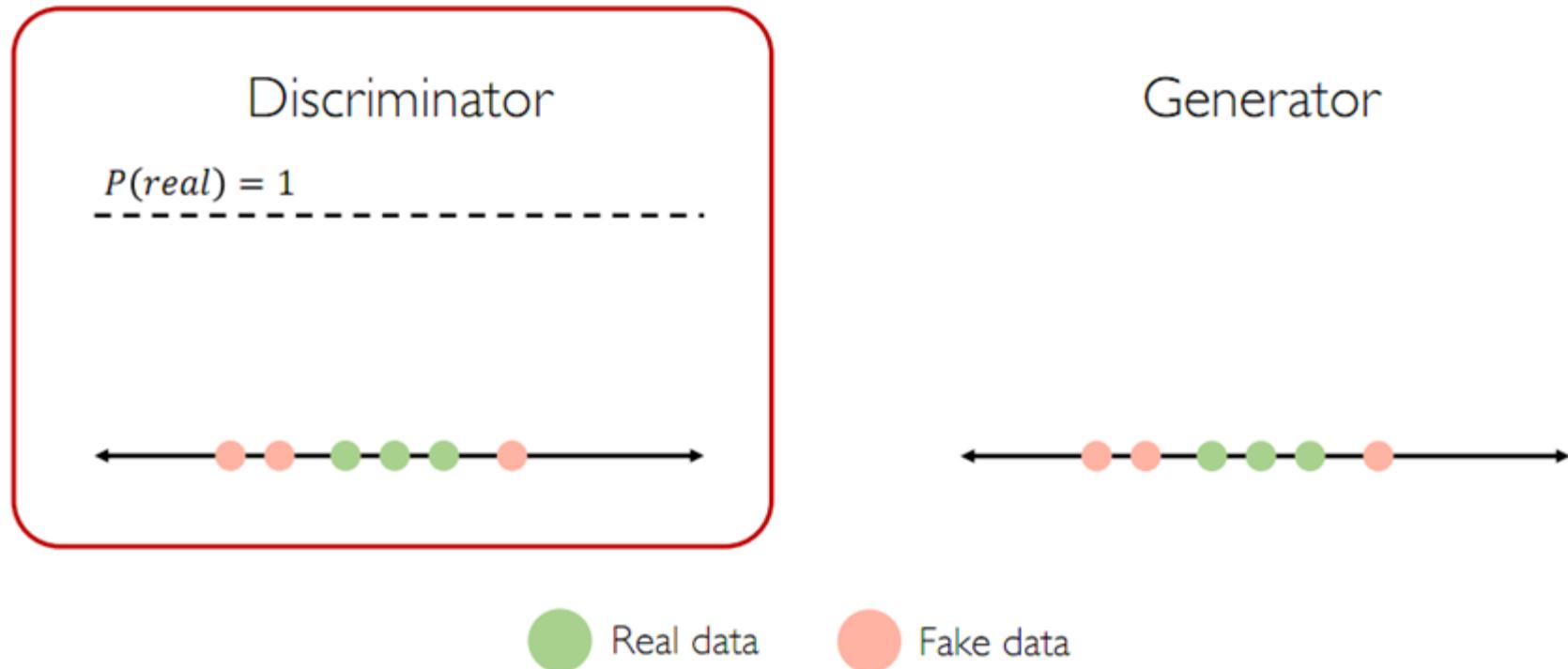
Trực giác ban đầu về GANs

- Generator cố gắng cải tiến chất lượng dữ liệu giả



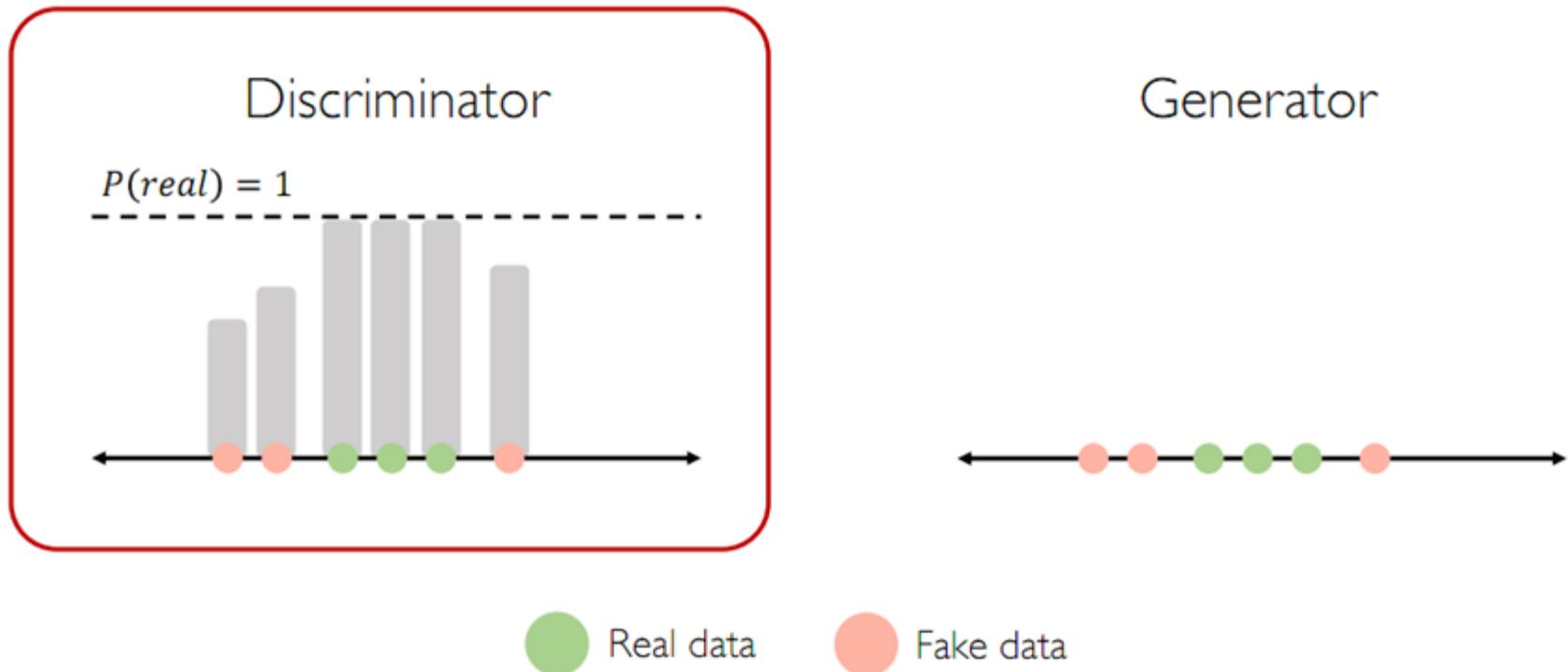
Trực giác ban đầu về GANs

- Discriminator đoán đâu là thật, đâu là giả



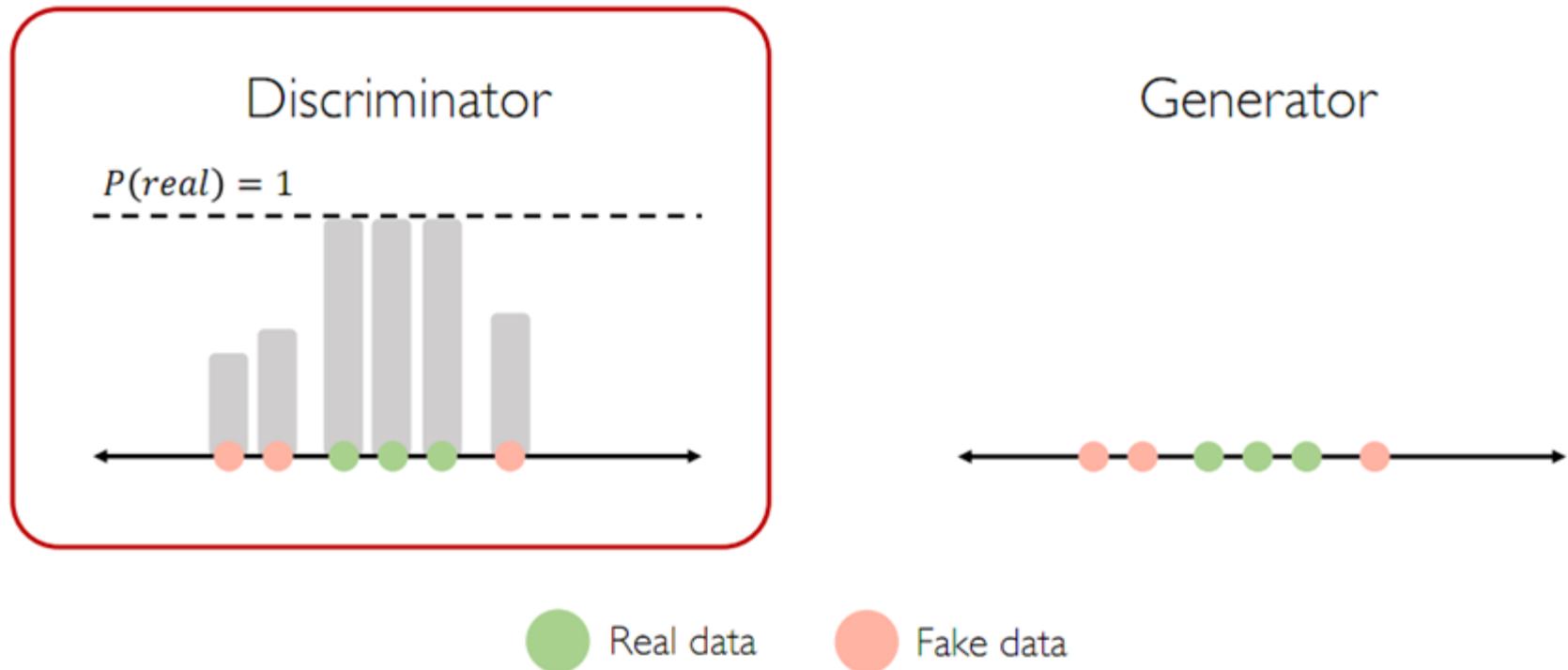
Trực giác ban đầu về GANs

- Discriminator đoán đâu là thật, đâu là giả



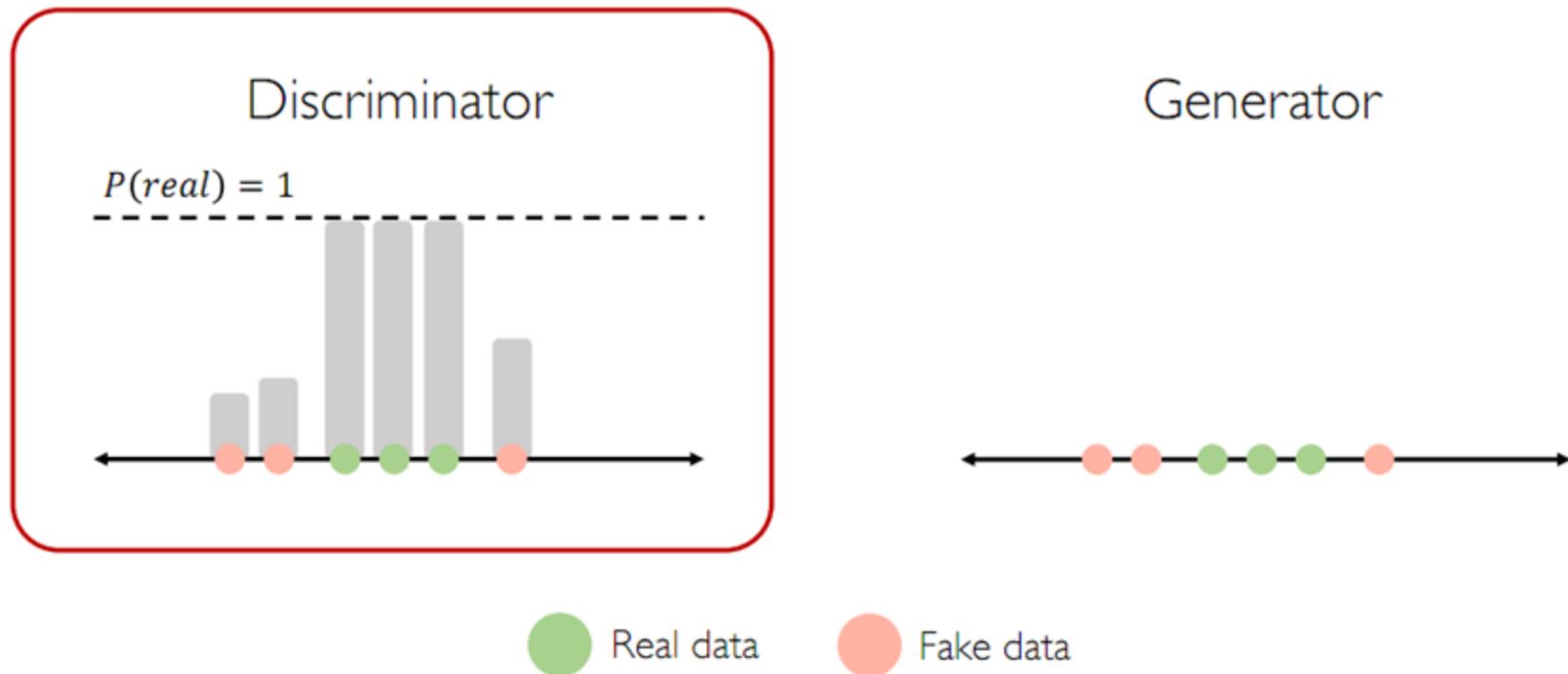
Trực giác ban đầu về GANs

- Discriminator đoán đâu là thật, đâu là giả



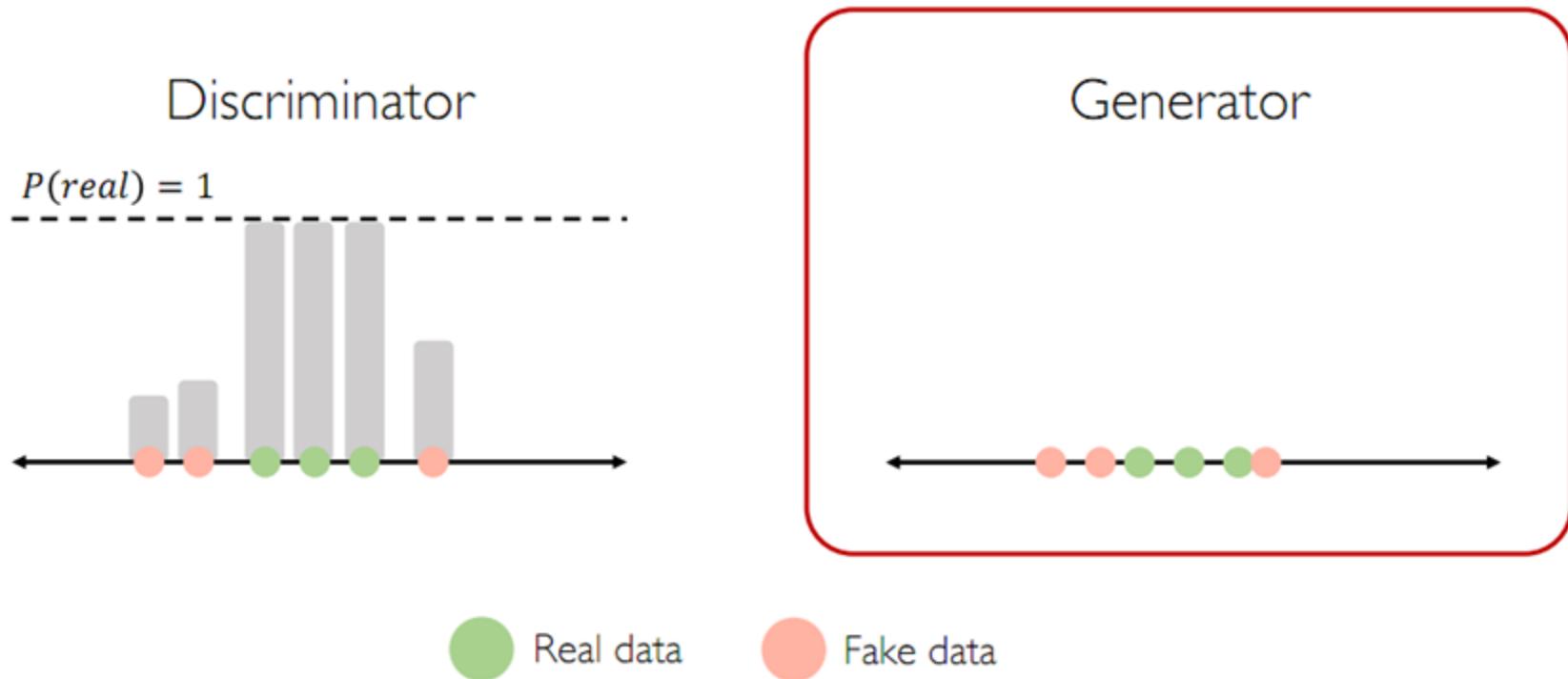
Trực giác ban đầu về GANs

- Discriminator đoán đâu là thật, đâu là giả



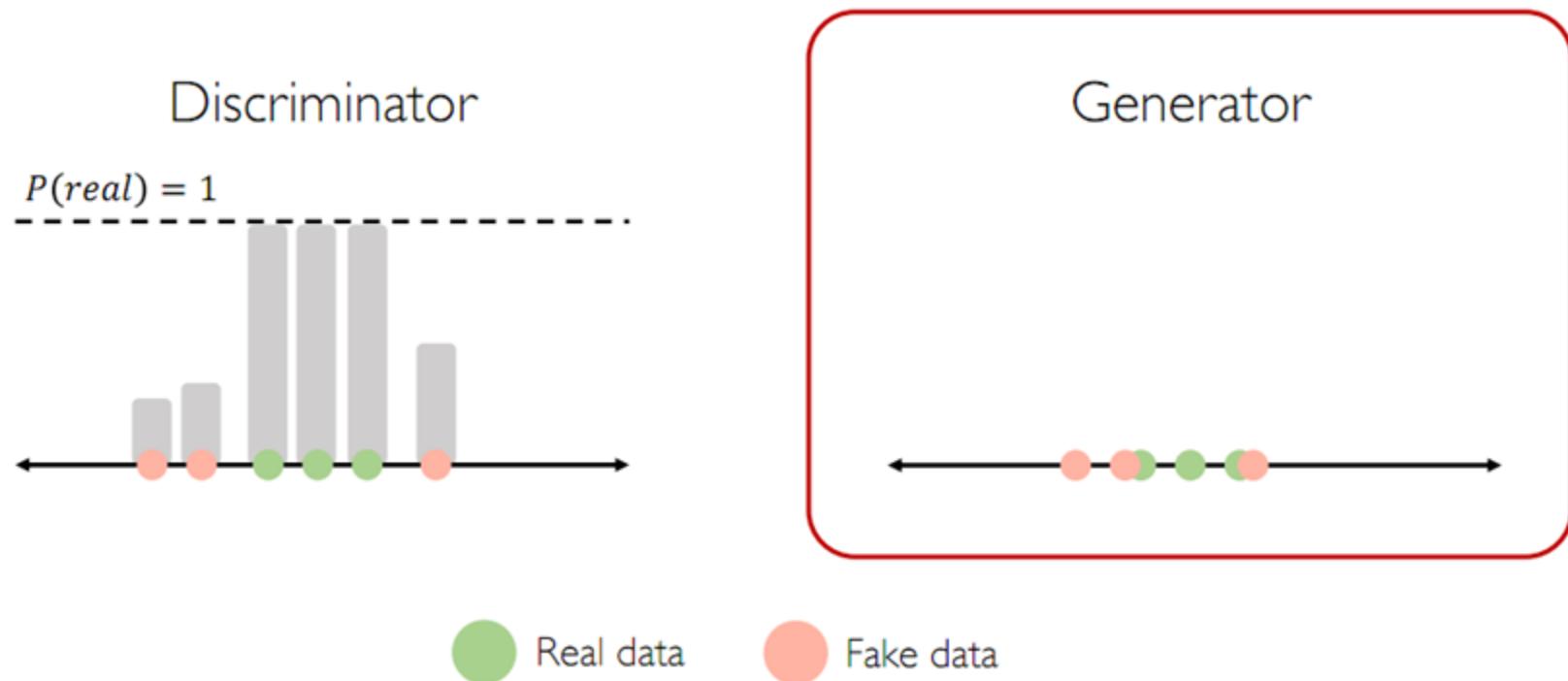
Trực giác ban đầu về GANs

- Generator cố gắng cải tiến chất lượng dữ liệu giả



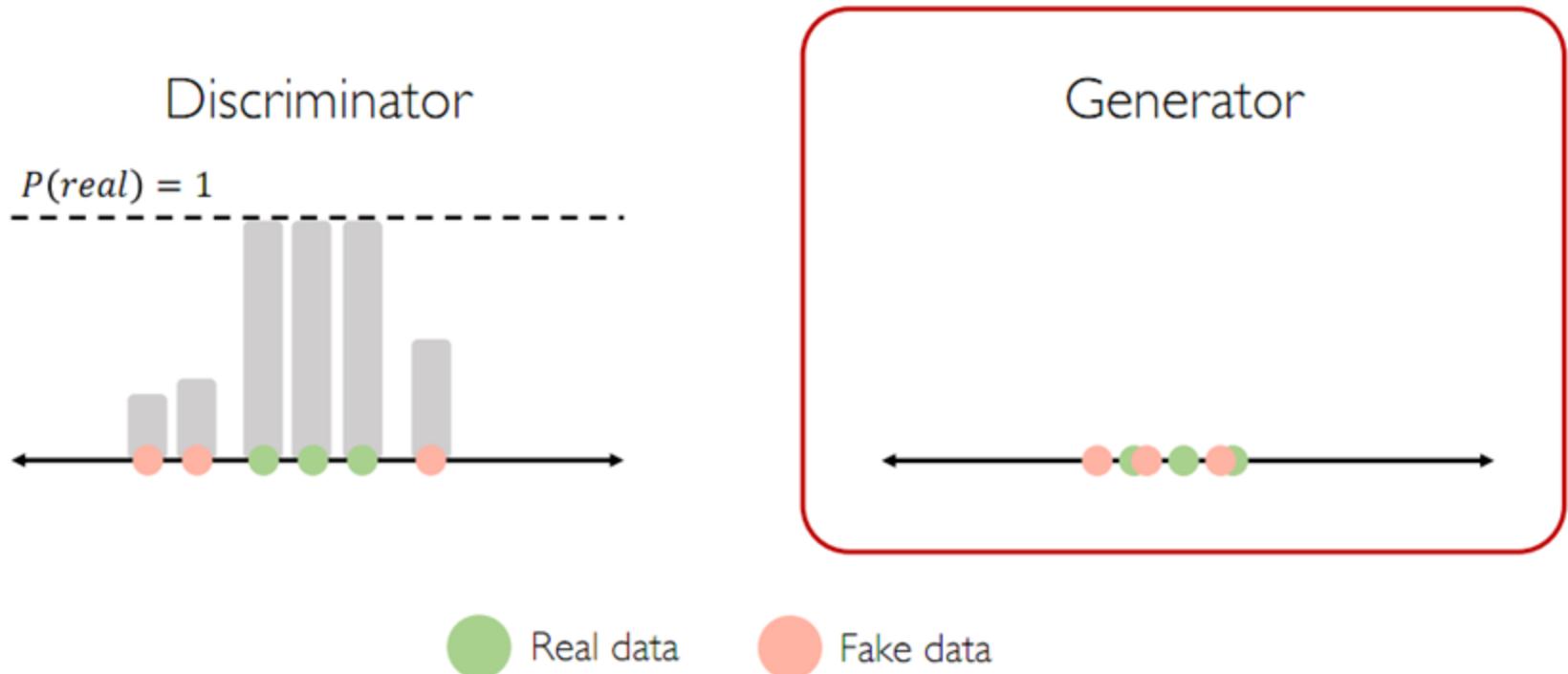
Trực giác ban đầu về GANs

- Generator cố gắng cải tiến chất lượng dữ liệu giả



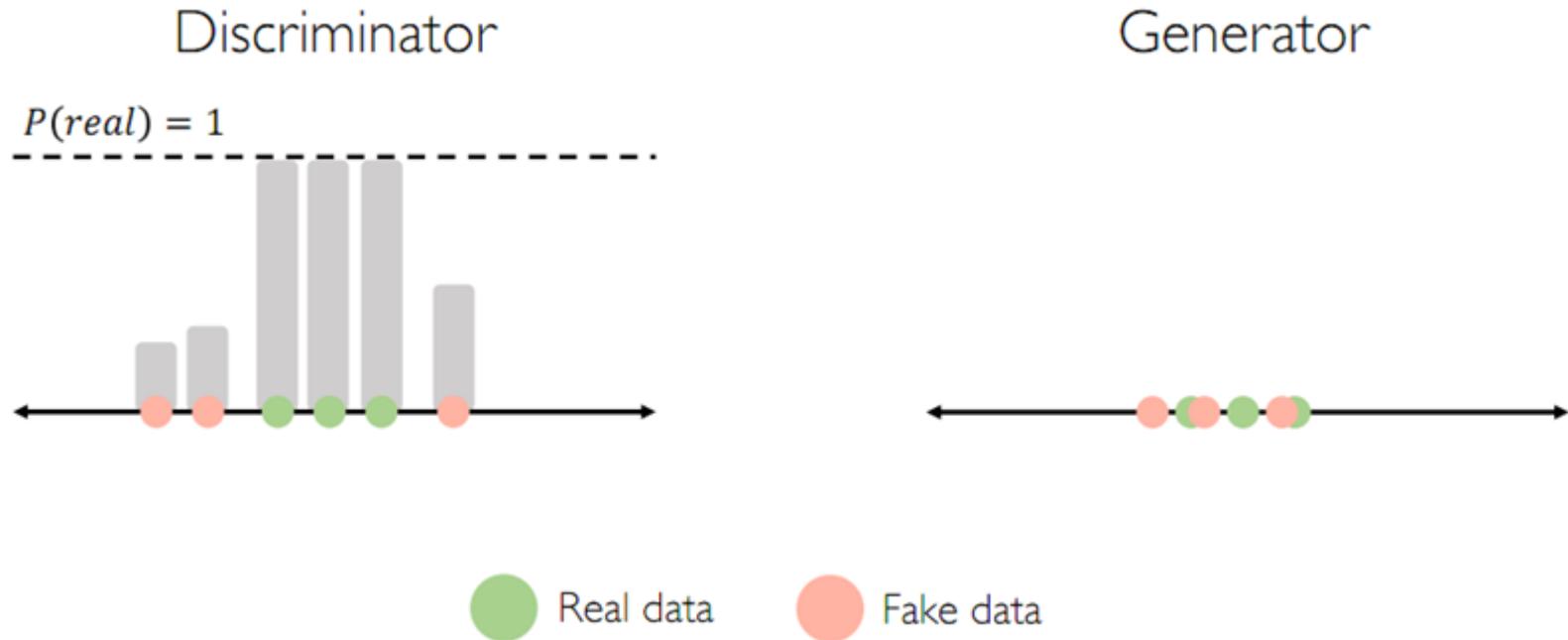
Trực giác ban đầu về GANs

- Generator cố gắng cải tiến chất lượng dữ liệu giả



Intuition behind GANs

- Discriminator cố gắng phân biệt dữ liệu thật và giả
- Generator cố gắng cải tiến chất lượng dữ liệu giả để lừa discriminator



Easy explanation

- A bad guy 1 want to make fake money for another bad guy 2 who handles the fake money



Easy explanation (cont)

- At first, fake money not good, bad guy 2: easy to recognize



Easy explanation (cont)

- Slowly better



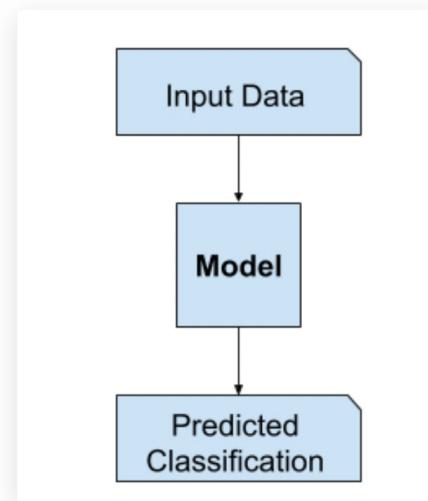
Easy explaination (cont)

- Finally



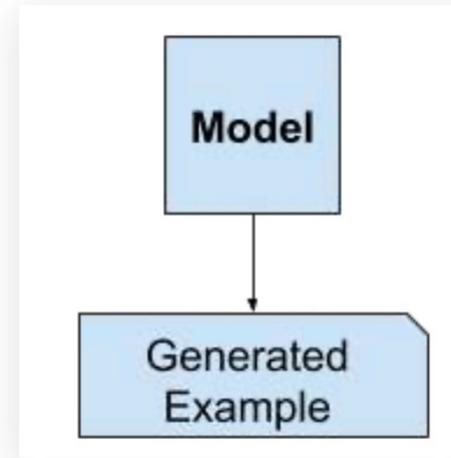
Discriminative model

- Quen thuộc: Classification → Discriminative model
 - Mục tiêu: tìm một hàm phân biệt $f(x)$ (*discriminant function*) để ánh xạ mỗi x lên một lớp nào đó



Generative model

- Mô hình tìm sự phân phối/ đặc trưng của dữ liệu đầu vào để tạo ra các dữ liệu mới tương đồng



Huấn luyện GANs

- Discriminator cố gắng phân biệt dữ liệu thật và giả
- Generator cố gắng cải tiến chất lượng dữ liệu giả để lừa discriminator

Train GAN jointly via **minimax** game:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log \left(1 - D_{\theta_d} \left(G_{\theta_g}(z) \right) \right) \right]$$

Discriminator wants to maximize objective s.t. $D(x)$ close to 1, $D(G(z))$ close to 0.

Generator wants to minimize objective s.t. $D(G(z))$ close to 1.

Huấn luyện GANs

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k , is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Sample minibatch of m examples $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\mathbf{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m \left[\log D \left(\mathbf{x}^{(i)} \right) + \log \left(1 - D \left(G \left(\mathbf{z}^{(i)} \right) \right) \right) \right].$$

end for

- Sample minibatch of m noise samples $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log \left(1 - D \left(G \left(\mathbf{z}^{(i)} \right) \right) \right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Kết quả lý thuyết

Proposition 1. For G fixed, the optimal discriminator D is

$$D_G^*(\mathbf{x}) = \frac{p_{data}(\mathbf{x})}{p_{data}(\mathbf{x}) + p_g(\mathbf{x})}$$

Proof:

$$\begin{aligned} V(G, D) &= \int_{\mathbf{x}} p_{data}(\mathbf{x}) \log(D(\mathbf{x})) d\mathbf{x} + \int_z p_{\mathbf{z}}(\mathbf{z}) \log(1 - D(g(\mathbf{z}))) dz \\ &= \int_{\mathbf{x}} p_{data}(\mathbf{x}) \log(D(\mathbf{x})) + p_g(\mathbf{x}) \log(1 - D(\mathbf{x})) d\mathbf{x} \end{aligned}$$

$$f(p_{data}, p_g) = p_{data}(x) \log(D(x)) + p_g(x) \log(1 - D(x)) dx$$

$$f' = p_{data}(x) \frac{1}{D(x) \ln(C)} - p_g(x) \frac{1}{(1 - D(x)) \ln(C)} dx = 0$$

$$D(x) = \frac{p_{data}}{p_{data} + p_g}$$

Kết quả lý thuyết

$$\begin{aligned} & \min_{\textcolor{brown}{G}} \max_{\textcolor{blue}{D}} \left(E_{x \sim p_{data}} [\log \textcolor{blue}{D}(x)] + E_{\textcolor{green}{z} \sim p(\textcolor{green}{z})} [\log (1 - \textcolor{blue}{D}(\textcolor{brown}{G}(\textcolor{green}{z})))] \right) \\ &= \min_{\textcolor{brown}{G}} \max_{\textcolor{blue}{D}} (E_{x \sim p_{data}} [\log \textcolor{blue}{D}(x)] + E_{x \sim p_{\textcolor{brown}{G}}} [\log (1 - \textcolor{blue}{D}(x))]) \\ &= \min_{\textcolor{brown}{G}} \int_X (p_{data}(x) \log \textcolor{blue}{D}_{\textcolor{brown}{G}}^*(x) + p_{\textcolor{brown}{G}}(x) \log (1 - \textcolor{blue}{D}_{\textcolor{brown}{G}}^*(x))) dx \\ &= \min_{\textcolor{brown}{G}} \int_X \left(p_{data}(x) \log \frac{p_{data}(x)}{p_{data}(x) + p_{\textcolor{brown}{G}}(x)} + p_{\textcolor{brown}{G}}(x) \log \frac{p_{\textcolor{brown}{G}}(x)}{p_{data}(x) + p_{\textcolor{brown}{G}}(x)} \right) dx \end{aligned}$$

Optimal Discriminator: $\textcolor{blue}{D}_{\textcolor{brown}{G}}^*(x) = \frac{p_{data}(x)}{p_{data}(x) + p_{\textcolor{brown}{G}}(x)}$

Kết quả lý thuyết

$$\begin{aligned} & \min_{\textcolor{brown}{G}} \max_{\textcolor{blue}{D}} \left(E_{x \sim p_{data}} [\log \textcolor{blue}{D}(x)] + E_{\textcolor{green}{z} \sim p(\textcolor{green}{z})} \left[\log (1 - \textcolor{blue}{D}(\textcolor{brown}{G}(\textcolor{green}{z}))) \right] \right) \\ &= \min_{\textcolor{brown}{G}} \int_X \left(p_{data}(x) \log \frac{p_{data}(x)}{p_{data}(x) + p_{\textcolor{brown}{G}}(x)} + p_{\textcolor{brown}{G}}(x) \log \frac{p_{\textcolor{brown}{G}}(x)}{p_{data}(x) + p_{\textcolor{brown}{G}}(x)} \right) dx \\ &= \min_{\textcolor{brown}{G}} \left(E_{x \sim p_{data}} \left[\log \frac{2}{2} \frac{p_{data}(x)}{p_{data}(x) + p_{\textcolor{brown}{G}}(x)} \right] + E_{x \sim p_{\textcolor{brown}{G}}} \left[\log \frac{2}{2} \frac{p_{\textcolor{brown}{G}}(x)}{p_{data}(x) + p_{\textcolor{brown}{G}}(x)} \right] \right) \\ &= \min_{\textcolor{brown}{G}} \left(E_{x \sim p_{data}} \left[\log \frac{2 * p_{data}(x)}{p_{data}(x) + p_{\textcolor{brown}{G}}(x)} \right] + E_{x \sim p_{\textcolor{brown}{G}}} \left[\log \frac{2 * p_{\textcolor{brown}{G}}(x)}{p_{data}(x) + p_{\textcolor{brown}{G}}(x)} \right] - \textcolor{purple}{\log 4} \right) \end{aligned}$$

Kết quả lý thuyết

$$\begin{aligned} & \min_{\textcolor{brown}{G}} \max_{\textcolor{blue}{D}} \left(E_{x \sim p_{data}} [\log \textcolor{blue}{D}(x)] + E_{\textcolor{green}{z} \sim p(\textcolor{green}{z})} \left[\log (1 - \textcolor{blue}{D}(\textcolor{brown}{G}(\textcolor{green}{z}))) \right] \right) \\ &= \min_{\textcolor{brown}{G}} \left(E_{x \sim \textcolor{red}{p}_{data}} \left[\log \frac{2 * p_{data}(x)}{p_{data}(x) + p_G(x)} \right] + E_{x \sim \textcolor{red}{p}_G} \left[\log \frac{2 * p_G(x)}{p_{data}(x) + p_G(x)} \right] - \log 4 \right) \\ &= \min_{\textcolor{brown}{G}} \left(KL \left(\textcolor{red}{p}_{data}, \frac{p_{data} + p_G}{2} \right) + KL \left(\textcolor{red}{p}_G, \frac{p_{data} + p_G}{2} \right) - \log 4 \right) \end{aligned}$$

Kullback-Leibler Divergence:

$$KL(\textcolor{red}{p}, \textcolor{violet}{q}) = E_{x \sim \textcolor{red}{p}} \left[\log \frac{\textcolor{red}{p}(x)}{\textcolor{violet}{q}(x)} \right]$$

Kết quả lý thuyết

$$\begin{aligned} & \min_{\textcolor{brown}{G}} \max_{\textcolor{blue}{D}} \left(E_{x \sim p_{data}} [\log \textcolor{blue}{D}(x)] + E_{\textcolor{green}{z} \sim p(\textcolor{green}{z})} \left[\log (1 - \textcolor{blue}{D}(\textcolor{brown}{G}(\textcolor{green}{z}))) \right] \right) \\ &= \min_{\textcolor{brown}{G}} \left(E_{x \sim p_{data}} \left[\log \frac{2 * p_{data}(x)}{p_{data}(x) + p_{\textcolor{brown}{G}}(x)} \right] + E_{x \sim p_{\textcolor{brown}{G}}} \left[\log \frac{2 * p_{\textcolor{brown}{G}}(x)}{p_{data}(x) + p_{\textcolor{brown}{G}}(x)} \right] - \log 4 \right) \\ &= \min_{\textcolor{brown}{G}} \left(KL \left(p_{data}, \frac{p_{data} + p_{\textcolor{brown}{G}}}{2} \right) + KL \left(p_{\textcolor{brown}{G}}, \frac{p_{data} + p_{\textcolor{brown}{G}}}{2} \right) - \log 4 \right) \\ &= \min_{\textcolor{brown}{G}} (2 * JSD(p_{data}, p_{\textcolor{brown}{G}}) - \log 4) \end{aligned}$$

JSD is always nonnegative, and zero only when the two distributions are equal!
Thus $p_{data} = p_G$ is the global min, QED

Jensen-Shannon Divergence:

$$JSD(\textcolor{red}{p}, \textcolor{violet}{q}) = \frac{1}{2} KL \left(\textcolor{red}{p}, \frac{\textcolor{red}{p} + \textcolor{violet}{q}}{2} \right) + \frac{1}{2} KL \left(\textcolor{violet}{q}, \frac{\textcolor{red}{p} + \textcolor{violet}{q}}{2} \right)$$

Kết quả lý thuyết

Theorem 1. *The global minimum of the virtual training criterion $C(G)$ is achieved if and only if $p_g = p_{\text{data}}$. At that point, $C(G)$ achieves the value $-\log 4$.*

Proof. For $p_g = p_{\text{data}}$, $D_G^*(\mathbf{x}) = \frac{1}{2}$, (consider Eq. 2). Hence, by inspecting Eq. 4 at $D_G^*(\mathbf{x}) = \frac{1}{2}$, we find $C(G) = \log \frac{1}{2} + \log \frac{1}{2} = -\log 4$. To see that this is the best possible value of $C(G)$, reached only for $p_g = p_{\text{data}}$, observe that

$$\mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} [-\log 2] + \mathbb{E}_{\mathbf{x} \sim p_g} [-\log 2] = -\log 4$$

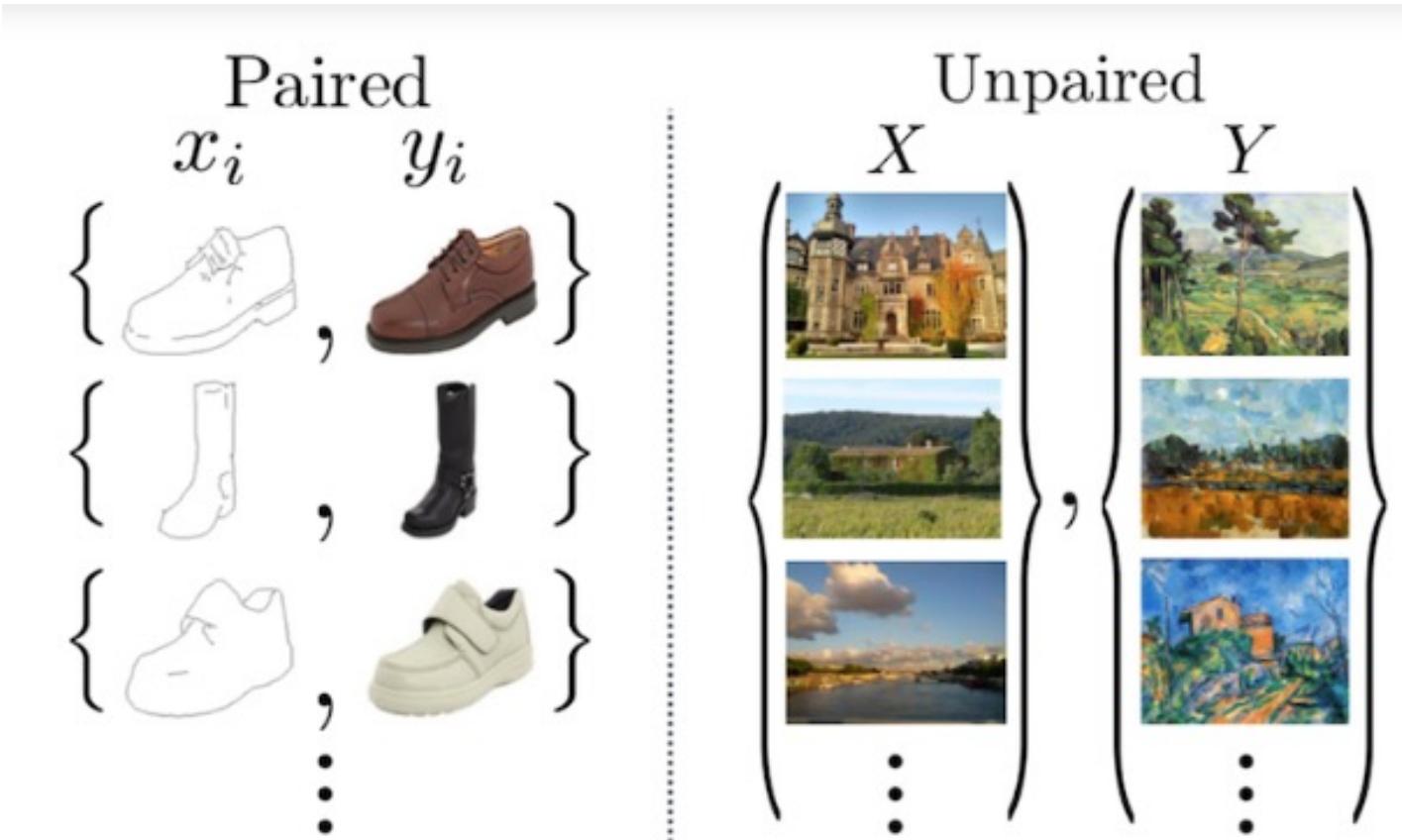
and that by subtracting this expression from $C(G) = V(D_G^*, G)$, we obtain:

$$C(G) = -\log(4) + KL \left(p_{\text{data}} \left\| \frac{p_{\text{data}} + p_g}{2} \right. \right) + KL \left(p_g \left\| \frac{p_{\text{data}} + p_g}{2} \right. \right) \quad (5)$$

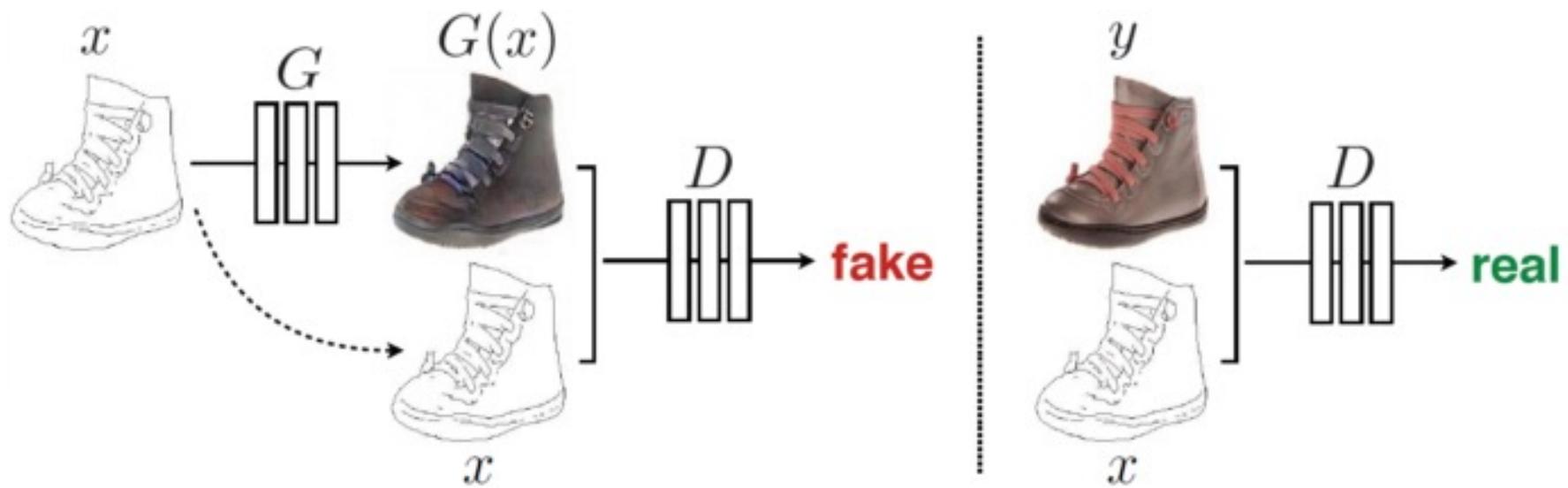
where KL is the Kullback–Leibler divergence. We recognize in the previous expression the Jensen–Shannon divergence between the model’s distribution and the data generating process:

$$C(G) = -\log(4) + 2 \cdot JSD(p_{\text{data}} \| p_g) \quad (6)$$

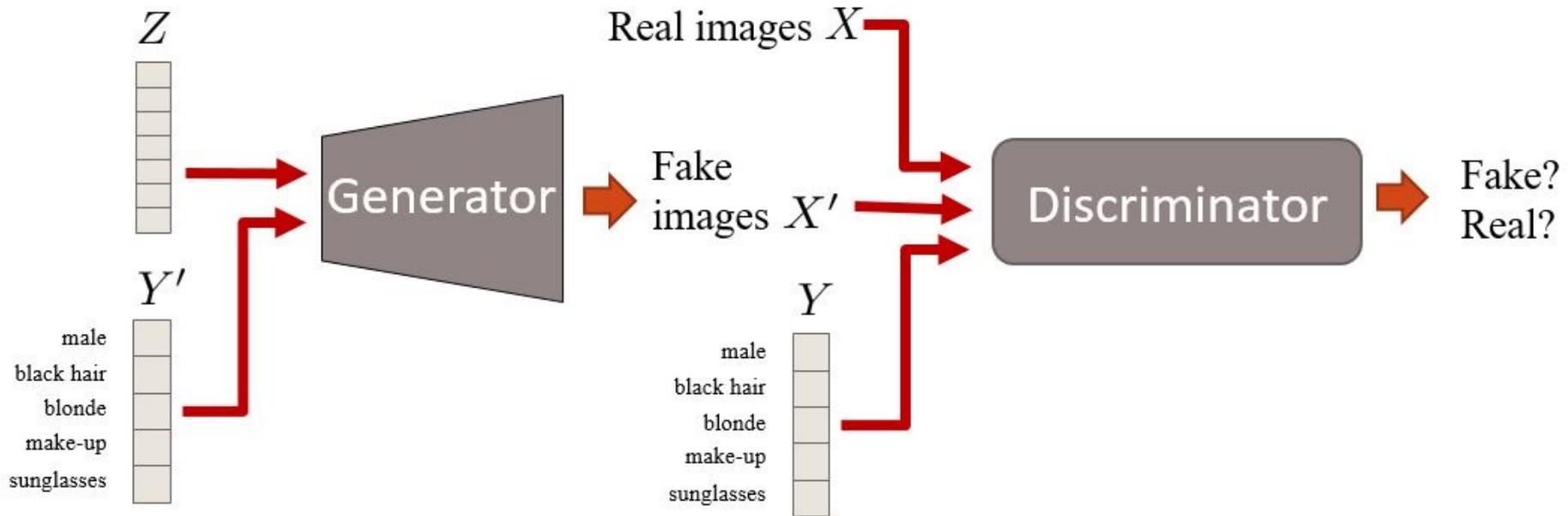
Biến đổi Image-to-image



pix2pix

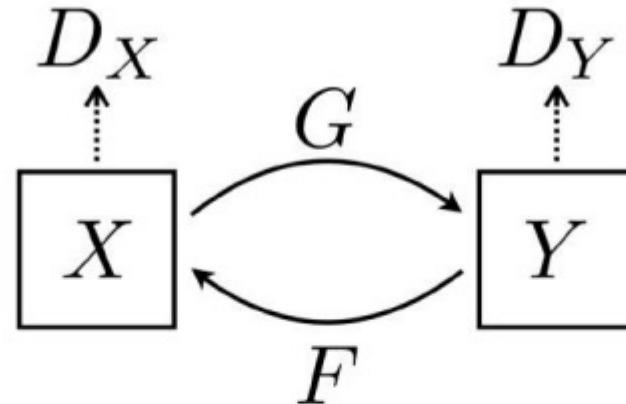


Conditional GAN

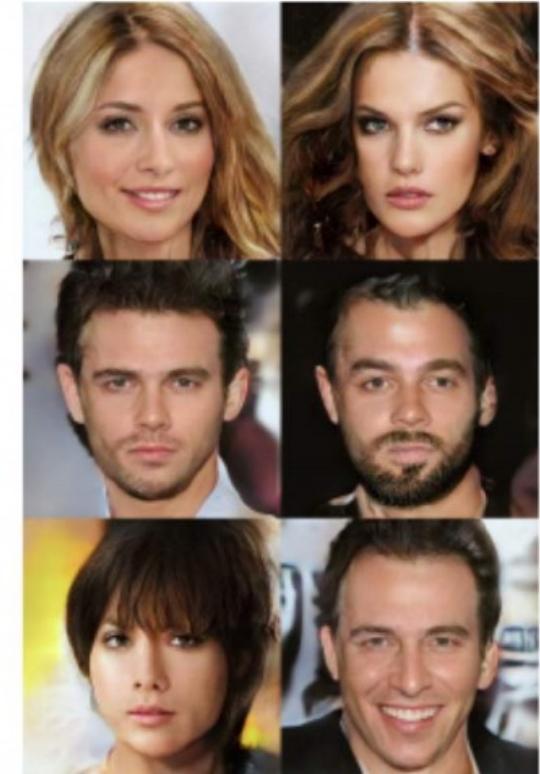
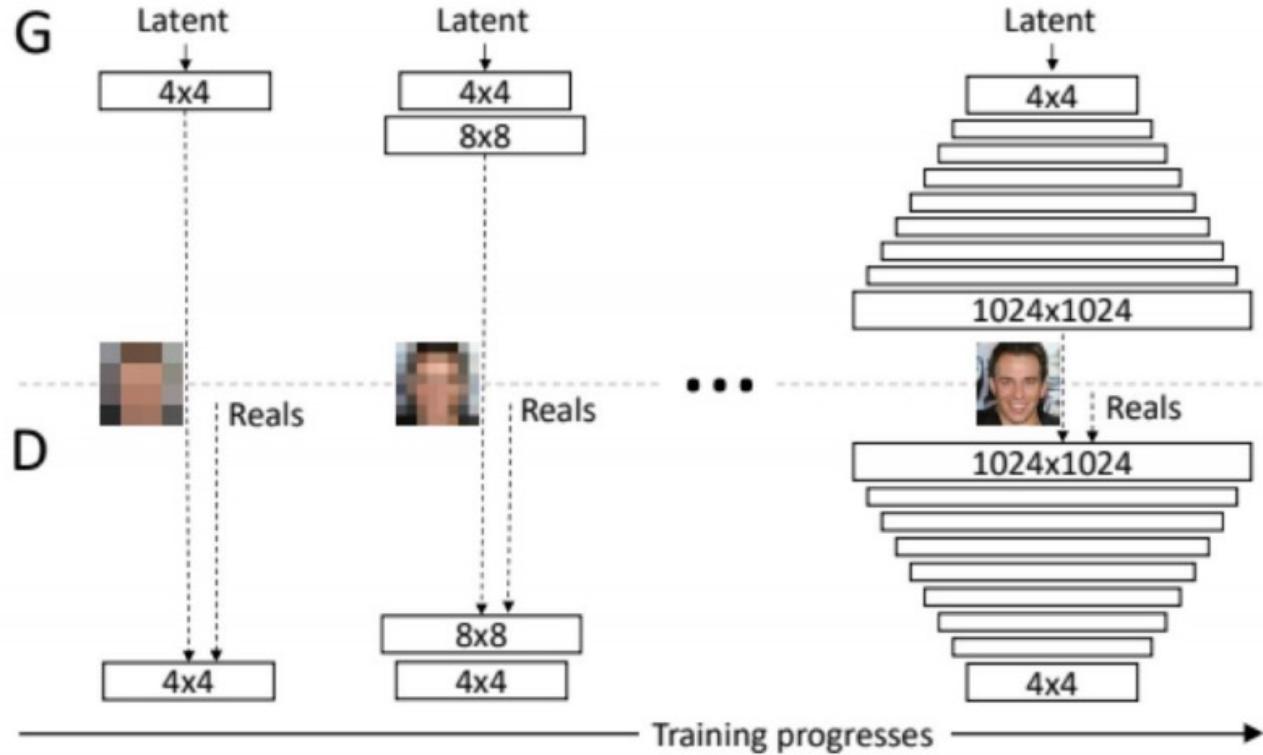


CycleGAN

CycleGAN learns transformations across domains with unpaired data.



Progressive growing of GANs (NVIDIA)



Karras et al., ICLR 2018.

Progressive growing of GANs (NVIDIA)



Karras et al., ICLR 2018.

Style-based generator



Karras et al., Arxiv 2018.

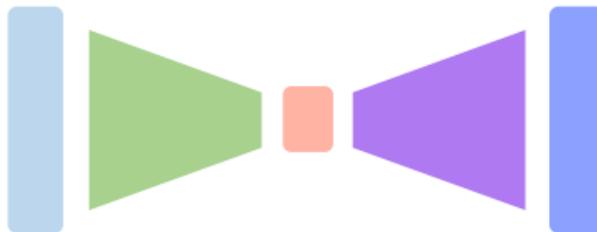
Style-based transfer



Tổng kết mô hình sinh

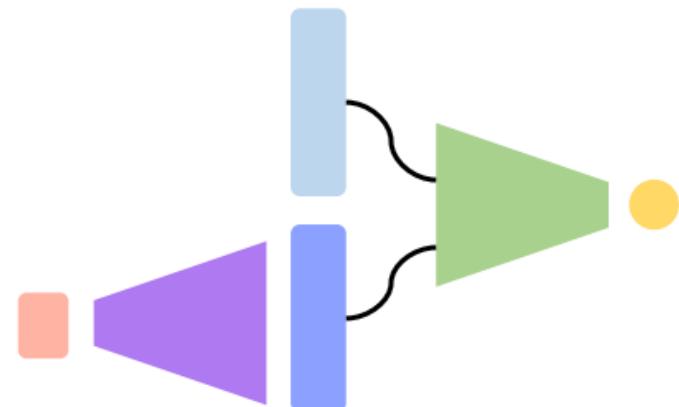
Autoencoders and Variational Autoencoders (VAEs)

Learn lower-dimensional latent space and **sample** to generate input reconstructions



Generative Adversarial Networks (GANs)

Competing **generator** and **discriminator** networks



Tài liệu tham khảo

1. Khóa MIT Deep Learning 6.S191:

<http://introtodeeplearning.com/>

2. Khóa cs231n của Stanford:

http://cs231n.stanford.edu/slides/2020/lecture_11.pdf



25
YEARS ANNIVERSARY
SOICT

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

**Thank you
for your
attentions!**

